



CENTRO UNIVERSITÁRIO DE BRASÍLIA - UniCEUB
Faculdades de Tecnologia e Ciências Sociais Aplicadas – FATECS
Curso de Engenharia da Computação

CIRCUITO RESIDENCIAL COM TEMPORIZADOR INDEPENDENTE POR USUÁRIO

RICARDO REBELO SILVA MELO
RA.: 2026811/3

Brasília/DF, Junho de 2008

CIRCUITO RESIDENCIAL COM TEMPORIZADOR INDEPENDENTE POR USUÁRIO

*Trabalho de conclusão de curso
apresentado à banca examinadora do
Centro Universitário de Brasília –
UniCEUB como requisito parcial para a
obtenção do Certificado de conclusão do
curso de Engenharia da Computação.
Prof. José Julimá Bezerra Júnior*

Brasília/DF.
2008.

DEDICATORIA

Dedico este projeto primeiramente a Deus, por este curso que foi conquistado com muita luta. A minha família, que sempre me apoiou e que faz parte desta minha vitória. A minha esposa, Liliane, pela paciência e motivação que me deu durante esta etapa da minha vida. Aos amigos, pela força e apoio.

AGRADECIMENTOS.

Agradeço aos meus amigos: Eliezer, pela força que me deu na construção do circuito eletrônico; Luciane, pelos finais de semanas sacrificados para que pudesse me ajudar no desenvolvimento do projeto; Luiz, pela paciência; mestre Sandoval, por noites perdidas de sono para que me ajudasse a entender o circuito e a programação; e Tiago Mitisuka, pela força e noites perdidas para que eu pudesse concluir esta fase da minha vida. Ao professor José Julimá, pelo apoio dado a esta batalha. Aos meus amigos Michel Calheiros, Leandro (Cabide), Leonardo (Frangolino), Rafael Ramos (Lineu) e os outros amigos por ter ajudado nas matérias e apoiado a terminar este curso. Em especial a minha esposa, Liliane, pelas noites em claro para que eu terminasse a minha monografia e o circuito. Aos meus pais pelo carinho, dedicação e amor nesta etapa tão importante da minha vida.

RESUMO

Este trabalho tem como objetivo principal criar uma forma de economizar energia em uma residência, utilizando um circuito microcontrolado. O circuito microcontrolado visa armazenar a senha de cada usuário e o tempo determinado para cada ambiente residencial. Por intermédio de um teclado, é possível digitar a senha para acessar o ambiente requerido. Através de um display, o usuário poderá acompanhar as informações solicitadas e verificar o tempo em que poderá permanecer no ambiente. Visando representar um circuito elétrico em um ambiente, uma lâmpada elétrica é utilizada como uma carga a ser acionada. Este tipo de controle pode ser estendido a outras aplicações, como, por exemplo, controle do tempo de banho quente utilizando um chuveiro elétrico. Generalizando, espera-se reduzir o consumo de energia, controlando o tempo de acionamento de um circuito elétrico.

Palavras-chave: microcontrolador, temporizador, *display*.

ABSTRACT

This monograph aims to create a way to save energy in residence using a circuit microcontroller. The circuit microcontroller aims to store the password for each user and the time given for each residential environment. Through a keypad, one can enter a password to access the required environment. Through a display, the user can monitor and verify the information requested as long as he remains in the environment. Aiming to represent an electrical circuit in an environment, an electric lamp is used as a load to be driven. This type of control can be extended to other applications, such as control of time using a hot bath, electric shower. Generalizing, it is expected to reduce energy consumption by controlling the time of trigger an electrical circuit.

Keywords: microcontroller, timer, display.

Sumário

ABSTRACT.....	7
LISTA DE FIGURAS.....	11
LISTA DE TABELAS	13
LISTA DE ABREVIATURAS E SIGLAS.....	14
1. INTRODUÇÃO.....	18
2. FUNDAMENTOS DE MICROPROCESSADORES.....	21
2.1. Representações de dados e do computador	21
2.2. Portas Lógicas	21
2.2.1. Chaves (switches) como Operadores Lógicos.....	21
2.2.2. Portas Básicas	24
2.3 Lógicas Sequenciais	25
2.3.1 Flip Flop	26
2.4 Dispositivos de Memória Semicondutores	26
2.4.1 RAM.....	26
2.4.2 ROM.....	27
2.5 Microprocessador.	27
2.6 Microcontrolador 8051 e as especificações do AT89S8252	30
2.6.1 Histórico do microcontrolador.....	30
2.6.2 Diferença entre Microcontrolador (MC) e Microprocessador (MP).....	30
2.6.3 Mundo Externo – Entrada/ Saída (I/O) e Periféricos	31
2.6.4 Microcontrolador 8051	32
2.6.5 Arquitetura Interna dos 8051	35
2.6.6 Descrição Formal da Pinagem do MC 8051	38

2.6.7	Organização das memórias	43
2.6.7.1	Memória para Código (CODE).....	43
2.6.7.2	Espaço para Manipulação de Dados (DATA).....	44
2.6.7.3	Registradores para Funções Especiais (Special Function Register – S.F.R).....	45
2.6.7.4	Extensão da Memória RAM (IDATA)	45
2.6.7.5	Extensão de Memória da Dados Externa (XData)	46
2.6.7.6	RAM Interna	46
2.6.7.7	Registradores Especiais Adicionais Presentes no AT89S8252	48
2.6.8	Clock e Reset	49
2.6.8.1	Clock.....	49
2.6.8.2	Reset	50
2.7	Registradores Especiais.....	51
2.7.1	Interrupções	52
2.7.1.1	Propriedades da interrupção.....	52
2.7.1.2	Interrupções na família 8051	53
2.8	Timer.....	56
2.8.1	Modos de trabalho do timer	56
2.8.2	Modo 0 (13 bits)	56
2.8.3	Modo 1 (16 bits).....	57
2.8.4	Modo 2 (8 bits com recarga automática).....	58
2.8.5	Modo 3 (8 bits mistos)	59
2.9	Serial	60
2.9.1	Modos de comunicação.....	60
2.9.2	O sistema de transmissão e recepção.	61
2.9.3	A palavra de controle <i>SCON</i>	62
2.10	Temporizador	62
3.	<i>Construção do Protótipo.....</i>	63

3.1. Hardware.....	64
3.1.1. Microcontrolador	64
3.1.2. Porta Serial	66
3.1.2.1. Comunicação com RS-232.....	66
3.1.2.2. Conversor USB-Serial	67
3.1.3. Circuito Integrado (CI) MAX 232	68
3.1.4. Circuito Integrado (CI) SN74HC126N.....	69
3.1.5. Display LCD de 2 linhas e 16 colunas.....	70
3.1.6. Teclado numérico matricial 4x3.	72
3.1.7. PC 817.	75
3.1.8. Relé.....	75
3.1.9. Notebook	76
3.2. Software.....	78
3.2.1. Sistema operacional	78
3.2.2. Proteus.	78
3.2.3. Terminal	79
3.2.4. Eclipse C/C++	80
3.2.5. MicroFlash.....	81
4. Conclusão.....	83
4.1. Resultado obtido	83
4.2. Dificuldades encontradas	83
4.3. Sugestões para trabalho futuros.....	85
5. Referencias Bibliográficas	86
Apêndice.....	88

LISTA DE FIGURAS

Figura 1.1 – Ilustração do circuito elétrico residencial.	19
Figura 2.1 – Uma chave controlando um <i>led</i>	22
Figura 2.2 – Duas chaves implementando uma função <i>OR</i> .	22
Figura 2.3 – Duas chaves implementando a função <i>AND</i>	23
Figura 2.4 – <i>Flip flop RS</i> (circuito, simbologia e tabela de funções)	26
Figura 2.5 – <i>Flip Flop D</i> (circuito, tabela de funções).	26
Figura 2.6 – Arquitetura básica do microprocessador	28
Figura 2.7 – Ilustração da <i>CPU</i> em ciclo de busca de instrução (na <i>ROM</i>).	29
Figura 2.8 - Diferença entre MC e MP	31
Figura 2.9 – Microcontrolador e seus periféricos de entrada e saída.	32
Figura 2.10 – Funcionamento do microcontrolador 8051	33
Figura 2.11 – Diagrama de interligação básica do microcontrolador 8051.	34
Figura 2.12 – Diagrama de bloco funcional do MC 8051	36
Figura 2.13 – Chip do 8051	38
Figura 2.14 - Representação das portas.	39
Figura 2.15 – Ilustração da porta P3	41
Figura 2.16 – Diferentes tipos de memórias do 8051	43
Figura 2.17 – Memória RAM.	47
Figura 2.18 – Ilustração de um ciclo de máquina no MC 8051.	50
Figura 2.19 – Reset automático e o Reset forçado	51
Figura 2.20 – Ilustração do processo de interrupção	52
Figura 2.21 – Processo de Interrupção não vetorada	54
Figura 2.22 – Interrupção na $\overline{INT0}$	55
Figura 2.23 – <i>TIMER/COUNTERS</i> em modo 0	57

Figura 2.24 – <i>TIMER/COUNTERS</i> em modo 1	58
Figura 2.25 – <i>TIMER/COUNTRS</i> em modo 2	58
Figura 2.26 – <i>TIMER/COUNTERS</i> em modo 3.	59
Figura 2.27 – Comunicação da <i>Serial</i>	60
Figura 2.28 – <i>SBUF</i> na <i>Serial</i>	61
Figura 2.29 – Controle da <i>SCON</i>	62
Figura 3.1 – Protótipo implementado	63
Figura 3.2 – Cabo conversor USB-Serial	68
Figura 3.3 – MAX 232	69
Figura 3.4 – SN74HC126N	70
Figura 3.5 – Display LCD 2X16	70
Figura 3.6 – Fluxograma do <i>Display</i>	71
Figura 3.7 – Teclado Numérico matricial 4x3	73
Figura 3.8 – Fluxograma do Teclado	74
Figura 3.9 – Foto acoplador PC817	75
Figura 3.10 – Foto do Relé	76
Figura 3.11 – Notebook	77
Figura 3.12 – Proteus 7.0	79
Figura 3.13 – Terminal V.25	80
Figura 3.14 – Eclipse C/C++	81
Figura 3.15 – <i>Microflash</i> .	82

LISTA DE TABELAS

Tabela 2.1 – Operação do circuito <i>OR</i>	23
Tabela 2.2 – operações do circuito <i>AND</i> .	23
Tabela 2.3 – portas lógicas básicas.	25
Tabela 2.4 – Comparação dos recursos dos MC citados.	37
Tabela 2.5 – Funções da porta P1.	40
Tabela 2.6 – Resumo das funções especiais da porta P3	42
Tabela 2.7 – Região <i>DATA</i>	45
Tabela 2.8 – Descrição de interrupção e o endereço de desvio de interrupção	54
Tabela 3.1 – Custo do protótipo	64
Tabela 3.2 - Pinos de comunicação Serial	66
Tabela 3.3 - Especificações do notebook	77

LISTA DE ABREVIATURAS E SIGLAS

A/D	<i>Analog/Digital</i> (Analógico/Digital)
ACC	Acumulador
AD	<i>Adress/Data</i> (Endereço/Dados)
ALE	<i>Address Latch Enable</i> (Ativar Endereço de Memória)
ALU	<i>Arithmeic-Logic Unit</i> (Unidade Lógica e Aritmética).
CI	Circuito Integrado
CPU	<i>Central Processing Unit</i> (Unidade Central de Processamento).
D/A	<i>Digital/Analog</i> (Digital/Analógico)
DVD	<i>Digital Video Disc</i> (Disco Vídeio Digital)
E/S	Entrada/Saída.
EA	Enable All (Permissão Total)
EA	<i>External Acess</i> (Acesso Externo)
EPROM	<i>Erasable Programmable Read-only Memory</i> (memória somente de leitura programável eletronicamente).
HD	<i>Hard Disc</i>
I/O	<i>INPUT/OUTPUT</i> (Entrada/Saída).
IE	<i>Interrupt Enable</i> (Permitir a interrupção)
IP	<i>Interrupt Priority</i> (Prioridade de Interrupção)
IR	<i>Recorder Instruction</i> (Registrador de Instrução)
MC	Microcontrolador
MP	Microprocessador
PC	<i>Personal Computer</i> (Computador Pessoal).
PSEN	<i>Program Store Enable</i> (Permitir Armazenar Programa)
RAM	<i>Random Access Memory</i> (memória de acesso aleatório).
ROM	<i>Ready Only Memory</i> (memória somente de leitura).
RS	<i>Recommended Standart</i> (Padrão Recomendado)
RST	Reset (Inicializar)
S.F.R	<i>Special Function Register</i> (Registradores para Funções Especiais)
	<i>UART Universal Asynchronous Receiver-Transmitter</i> (Transmissor e receptor assíncrono universal).
Vcc	<i>Collector Common Voltage</i> (Coletor Comum de Tensão)

1. INTRODUÇÃO

Este trabalho tem por finalidade apresentar um circuito que monitore o tempo em que o usuário utiliza a energia em uma residência através de um temporizador. Especificamente, é utilizado um microcontrolador, onde serão registrados os usuários e suas respectivas senhas. Cada usuário registrado terá uma senha que determinará um tempo de utilização dos circuitos elétricos residenciais. Este tempo é mostrado em um display. Depois de esgotado o tempo, a energia é desligada e o usuário não terá tempo adicional. Ciente disso, o usuário deverá adequar a utilização da energia ao tempo que lhe é disponibilizado.

Neste trabalho, uma lâmpada convencional é utilizada representando um circuito elétrico residencial. Esta lâmpada é montada com os circuitos elétricos. Na apresentação, a senha de cada usuário deve ser digitada em um teclado numérico. Esta senha é armazenada no microcontrolador e, ao ser validado, a lâmpada é ligada. A lâmpada ligada demonstra que está recebendo corrente e, assim, o *display* começará a marcar o tempo de utilização para aquele ambiente residencial. Se a lâmpada estiver desligada, significa que não há corrente e, portanto, o *display* não irá marcar o tempo daquele ambiente residencial.

1.1. Motivação e posicionamento

Diante dos atuais problemas mundiais de energia, este projeto pretende mostrar uma maneira de economizar energia em ambientes residenciais e comerciais através de um temporizador controlado por senha.

A solução deste problema é baseada em conhecimentos de microcontroladores programáveis, interfaces com o circuito de energia elétrica que alimenta a energia residencial e base de tempo para que uma pessoa utilize a energia em cada ambiente residencial. Espera-se com isto mostrar que seja possível economizar energia apenas adequando e controlando o tempo de utilização de energia no ambiente residencial.

1.2. Objetivos

O objetivo geral é economizar energia elétrica em uma residência através de um temporizador microcontrolado. Um microcontrolador é utilizado para registrar e armazenar a senha de cada usuário. Cada usuário terá um tempo de utilização dos circuitos elétricos residências. O usuário poderá acompanhar em um *display* o tempo que foi determinado para a utilização da energia. Após este tempo a energia será desligada e o usuário não terá tempo adicional.

1.3. Visão Geral do projeto

O projeto é composto por microcontrolador, relé, lâmpada, um teclado matricial 3x4, um *display* 2x16 e de um acionamento. O microcontrolador terá uma senha *default* já registrada na memória interna do mesmo. A figura 1.1 mostra o digrama do circuito.

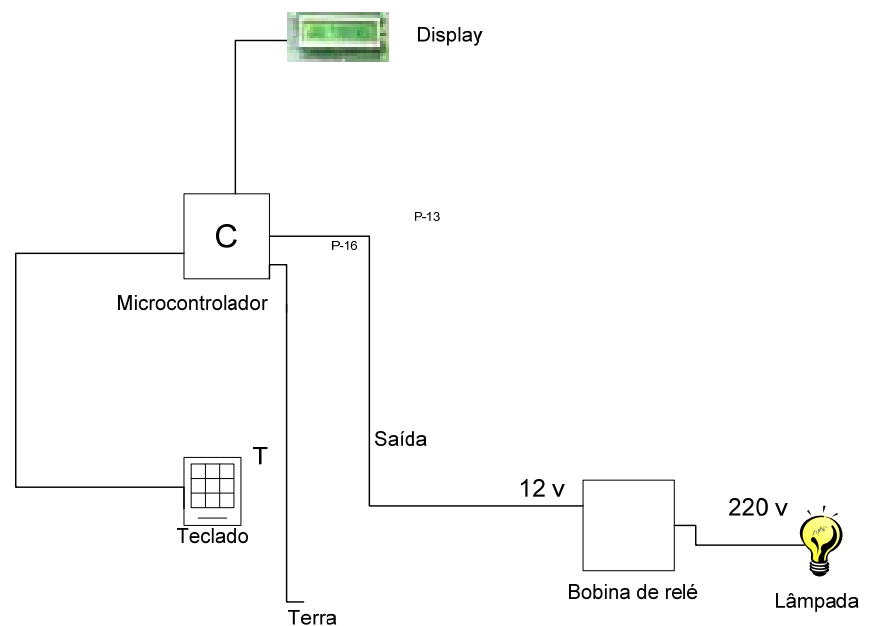


Figura 1.1 – Ilustração do circuito elétrico residencial.

Para que o usuário controle a energia em um ambiente residencial é necessário que o mesmo digite a senha *default*, já armazenada no microcontrolador. Ao digitar a senha corretamente, a senha que está armazenada no microcontrolador, mostrará no *display* uma mensagem ao usuário para que ele determine o tempo para aquele ambiente residencial. O tempo ao ser determinado pelo o usuário começara a decrementar até o zerar, após o ser zerado os equipamentos eletrônicos e a parte elétrica daquele ambiente residencial serão desligados.

O principal componente utilizado neste projeto para o funcionamento do *display* e do teclado é o microcontrolador da família 8051, modelo AT89S8252, além dos componentes auxiliares. O componente acionado é composto de relé de entrada 5 volts e saída de 220v e um foto acoplador, PC186 , que tem como função acionar a lâmpada, utilizada para simular um ambiente residencial.

Além deste, esta monografia está dividida em mais três capítulos.

O capítulo 2 trata sobre os principais tópicos teóricos utilizados.

O capítulo 3 explica o desenvolvimento do projeto. Este capítulo detalha o funcionamento do protótipo na parte do hardware e do software.

O capítulo 4 explica a conclusão do projeto. Neste capítulo são colocadas as principais dificuldades encontradas e a sugestões futuras para este trabalho.

2. FUNDAMENTOS DE MICROPROCESSADORES

2.1. Representações de dados e do computador

“Os computadores arquivam, manipulam e transformam informações, que são chamadas de dados (*data*). Dados são representados normalmente por textos ou números. Antes de arquivar, manipular e transformar estes dados ou informações é necessário um sistema que o represente.” (NICOLOSI, 2004, p.4)

No sistema computacional utiliza-se o sistema binário para a comunicação entre os componentes, estes símbolos são “0” e “1”. Normalmente, “0” é armazenado como baixa tensão (0 volts) e o “1” é armazenado como alta tensão (5 volts).

O dígito binário simples é chamado de *bit*. O *bit* é armazenado em um dispositivo eletrônico chamado de *flip-flop*, que tem como capacidade a indicação da posição de cada *bit*.

Os computadores e microcontroladores trabalham com dados de 8 *bits*. O termo *bytes* veio justamente para descrever este tipo de dados. Veja:

1 *Byte* = 8 *bits*.

2 *Bytes* = 16 *bits* - Word (Palavra).

1 *Kbyte* = 1024 *bytes* = 8192 *bits*.

1 *Mbyte* = 1024 *Kbytes* = 1048576 *bytes* = 8388608 *bits*.

A representação de números negativos é chamada de *sign-magnitude*. Usando o *sign-magnitude*, o *bit* da esquerda de um número binário representa o *bit* sinal. Se o *bit* for 1, o número é negativo. Se *bit* for 0, o número é positivo.

2.2. Portas Lógicas

2.2.1. Chaves (switches) como Operadores Lógicos

O circuito lógico é implementado por um circuito simples controlado por uma chave (*switch*) e um diodo emissor de luz (*led*).

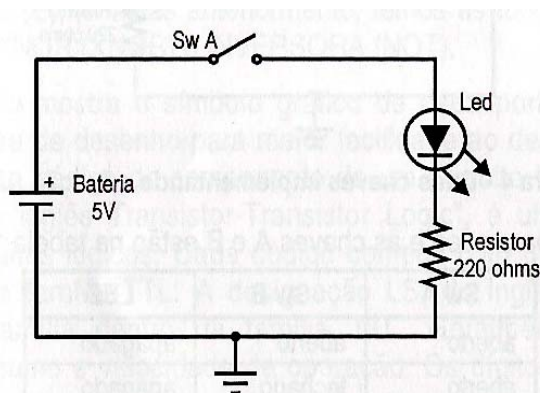


Figura 2.1 – uma chave controlando um *led*. (NICOLSI, 2000, p.23)

A figura 2.1 demonstra que quando uma determinada corrente percorre pelo circuito o *led* se acende, caso contrário, o *led* se apaga. A chave *Sw A* tem a função de ligar e desligar o circuito. Quando a chave *Sw A* estiver aberta, não terá corrente no circuito, assim o *led* permanecerá desligado. Quando esta chave estiver fechada, terá corrente no circuito e, conseqüentemente, o *led* permanecerá aceso.

Os tipos mais comuns de funções lógicas são: função lógica *OR* e função lógica *AND*.

A função lógica *OR* está demonstrada na figura 2.2:

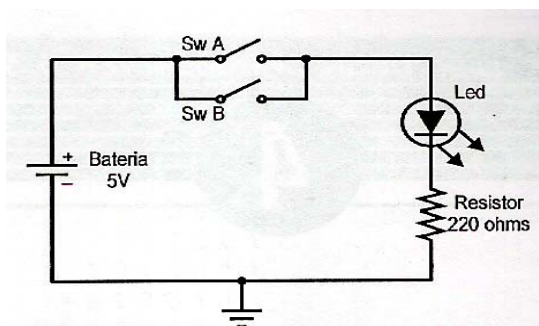


Figura 2.2 – duas chaves implementando uma função *OR*. (NICOLSI, 2000, p.24).

A figura 2.3 demonstra uma ligação em paralelo, ou seja, se a chave *Sw A* ou a chave *Sw B* estiverem fechadas, o *led* acenderá. A tabela 1 mostra a

possíveis posições das chaves e a correspondente condição do *led*. A equação utilizada na função *OR* é: $X = A \vee B$, $X = A + B$.

Sw A	Sw B	Led
Aberto	Aberto	Apagado
Aberto	Fechado	Aceso
Fechado	Aberto	Aceso
Fechado	Fechado	Aceso

Tabela 2.1 – Operação do circuito *OR*. (NICOLSI, 2000, p.24).

A função lógica *AND* está demonstrada na figura 4:

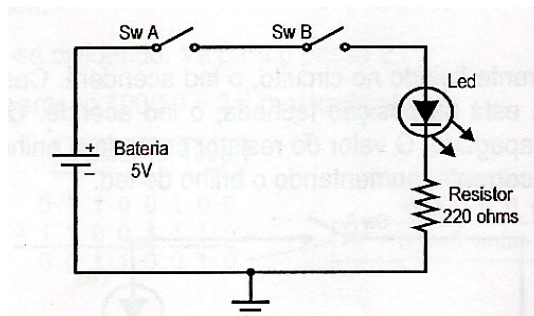


Figura 2.3 – duas chaves implementando a função *AND*. (NICOLSI, 2000, p.24).

A figura 2.3 demonstra uma ligação em série, ou seja, para que o *led* acenda, é necessário que a chave *Sw A* e a chave *Sw B* estejam fechadas. Se algumas destas chaves estiverem abertas o *led* estará apagado. A tabela 2.2 mostra a possíveis posições das chaves e a condição do *led*. A equação utilizada na função *AND* é: $X = A \wedge B$ ou $X = A . B$.

Sw A	Sw B	Led
Aberto	Aberto	Apagado
Aberto	Fechado	Apagado
Fechado	Aberto	Apagado
Fechado	Fechado	Aceso


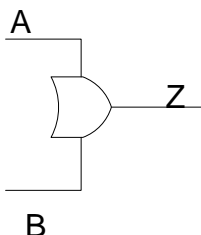
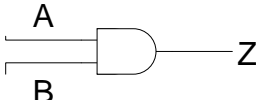
Tabela 2.2 – Operações do circuito *AND*. (NICOLSI, 2000, p.24).

A operação matemática que se dá através do circuito lógico é chamada de Álgebra Booleana. Este circuito lógico é de fácil interpretação. Quando o *led* estiver aceso, corresponde ao número binário 1. Caso esteja apagado, significa o número zero. A chave pode ser descrita utilizando o número 1 para a chave aberta e 0 para a chave fechada.

2.2.2. Portas Básicas

Existem outros tipos de funções lógicas, tais como: *NAND*, *NOR*, *EXOR* (*XOR*, *OU - eXclusivo*), *EXNOR* (*XNOR*) e *INVERSORA* (*NOT*).

A tabela 2.3 demonstra todas as funções lógicas, símbolos, TTL¹, equações e tabela-verdade.

Operações	Símbolo	TTL	Equações	Tabela -Verdade															
Inversora ou NOT		74LS04	$Z = \overline{A}$	<table><tr><th>A</th><th>Z</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	A	Z	0	1	1	0									
A	Z																		
0	1																		
1	0																		
OR		74LS32	$Z = A+B$ $Z = A \vee B$	<table><tr><th>A</th><th>B</th><th>Z</th></tr><tr><td>1</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>0</td><td>0</td><td>0</td></tr></table>	A	B	Z	1	1	1	1	0	1	0	1	1	0	0	0
A	B	Z																	
1	1	1																	
1	0	1																	
0	1	1																	
0	0	0																	
AND		74LS08	$Z = A \cdot B$ $Z = A \wedge B$	<table><tr><th>A</th><th>B</th><th>Z</th></tr><tr><td>1</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td></tr></table>	A	B	Z	1	1	1	1	0	0	0	1	0	0	0	0
A	B	Z																	
1	1	1																	
1	0	0																	
0	1	0																	
0	0	0																	

¹ TTL – Transistor-Transistor Logic – Circuito utilizado para implementação de circuito lógico. Cada código começa com o número 74 para indicar que o circuito Integrado (CI) é membro da família TTL. O LS - Low Power Schottky - quer dizer a subfamília do TTL. Esta subfamília é diferenciada pelo consumo e velocidade de operação. Os últimos dígitos se referem à função do dispositivo.

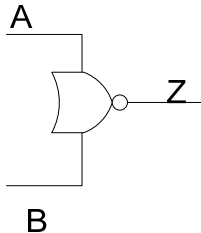
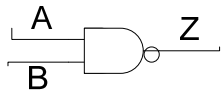
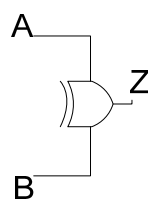
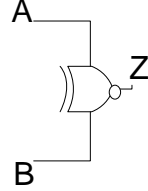
NOR		74LS02	$Z = \overline{A+B}$ $Z = \overline{A \vee B}$	<table><tr><th>A</th><th>B</th><th>Z</th></tr><tr><td>1</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td></tr></table>	A	B	Z	1	1	0	1	0	0	0	1	0	0	0	1
A	B	Z																	
1	1	0																	
1	0	0																	
0	1	0																	
0	0	1																	
NAND		74LS00	$Z = \overline{A \cdot B}$ $Z = \overline{A \wedge B}$	<table><tr><th>A</th><th>B</th><th>Z</th></tr><tr><td>1</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>0</td><td>0</td><td>1</td></tr></table>	A	B	Z	1	1	0	1	0	1	0	1	1	0	0	1
A	B	Z																	
1	1	0																	
1	0	1																	
0	1	1																	
0	0	1																	
EXOR		74LS86	$Z = A(+)B$ $Z = A \setminus / B$	<table><tr><th>A</th><th>B</th><th>Z</th></tr><tr><td>1</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>0</td><td>0</td><td>0</td></tr></table>	A	B	Z	1	1	0	1	0	1	0	1	1	0	0	0
A	B	Z																	
1	1	0																	
1	0	1																	
0	1	1																	
0	0	0																	
EXNOR		74LS66	$Z = \overline{A(+)B}$ $Z = \overline{A \setminus / B}$	<table><tr><th>A</th><th>B</th><th>Z</th></tr><tr><td>1</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td></tr></table>	A	B	Z	1	1	1	1	0	0	0	1	0	0	0	1
A	B	Z																	
1	1	1																	
1	0	0																	
0	1	0																	
0	0	1																	

Tabela 2.3 – portas lógicas básicas. (NICOLOSI, 2000, p.26)

Utilizam-se as equações referidas anteriormente nas funções *OR* e *AND*. O uso do traço em cima refere-se à negação ou inversão lógica.

2.3 Lógicas Seqüenciais

A lógica seqüencial é uma operação determinada por um pulso de *clock* de sincronismo em um circuito lógico.

2.3.1 Flip Flop

O *flip flop* ou *latch* é uma célula de memória, ou seja, uma célula lógica capaz de manter um estado após a mudança na entrada. Este estado conservado é como se fosse gravado.

A figura 2.4 mostra o *flip flop RS*.

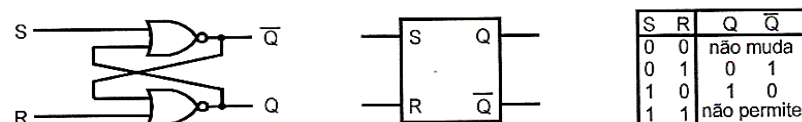


Figura 2.4 – *Flip flop RS* (circuito, simbologia e tabela de funções). (NICOLSI, 2000, p.48)

O *flip flop RS* tem duas entradas que são: S para *Set* e R para *Reset*; além de duas saídas que são: Q e seu complemento \overline{Q} . O Q = 1 quer dizer que o *flip flop* foi setado, e o \overline{Q} = 0 quer dizer que o *flip flop* resetado.

A figura 2.5 mostra o *flip flop D*.

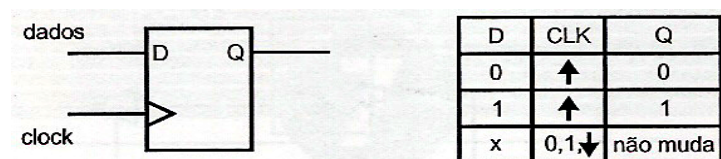


Figura 2.5 – *Flip Flop D* (circuito, tabela de funções). (NICOLSI, 2000, p.49)

O *flip flop D* tem uma entrada, um *clock* e uma saída. O *clock* é representado por um símbolo que é a seta: “seta para cima” significa uma transição 0-para-1, borda positiva ou borda de subida na entrada do *clock*; “seta para baixo” representa uma transição 1-para-0, ou borda negativa, ou borda de descida na entrada do *clock*. Se no *clock* de entrada o símbolo for um triângulo implica dizer que ele é ativo na borda de subida. Todavia, se o símbolo for uma circunferência, expressa um *clock* ativo em borda de descida.

2.4 Dispositivos de Memória Semicondutores

2.4.1 RAM

A palavra *RAM* significa *Random Access Memory* (Memória de Acesso Aleatório). A memória *RAM* contém centenas de células, em alguns casos

podendo ter milhares de células. A memória *RAM* possui outro tipo de memória que é a *DRAM*, ou *RAM* dinâmica. Esta memória serve para escrita na memória.

2.4.2 ROM

A palavra *ROM* significa *Read Only Memory*, ou memória apenas de leitura de dados. Outros tipos de memória *ROM* são: *PROM* – *ROM* programável são programáveis pelo usuário por meio de hardware especial. As memórias *Erasable PROM*, *EPROM*, *PROM* apagável permitem apagar os dados, pela exposição do dispositivo à luz ultravioleta. Já as memórias *Electrically Erasable PROM*, *EEPROM*, *E²PROM*, *PROM* “apagável” eletricamente permitem que os dados sejam apagados eletricamente.

2.5 Microprocessador.

O microprocessador é um elemento eletrônico, desenvolvido para executar tarefas específicas, com linguagens de comando específico. Ele utiliza uma memória *ROM* para leitura de dados e uma memória *RAM* para armazenamento temporário de dados.

Os microprocessadores são utilizados em equipamentos eletrônicos digitais e analógicos, sendo mais comum em equipamentos eletrônicos digitais.

O microprocessador tem como tarefa específica gravar em sua memória de código, a “*ROM*”, que por sua vez, tem o poder de “ler” uma tecla de teclado, acender uma lâmpada, “escrever” em uma impressora, “comunicar-se” com um “*PC*”, ligar/desligar motores, entre outros.

A programação em *ROM* deve ser em códigos compatíveis com a linguagem do microprocessador, para que possam ser executados uma tarefa ou grupo de tarefas. O projetista deve entender a parte lógica da linguagem da máquina (*C*, *Assembler*) e a parte física que é o hardware.

A figura 2.6 mostra a Arquitetura Básica do Microprocessador.

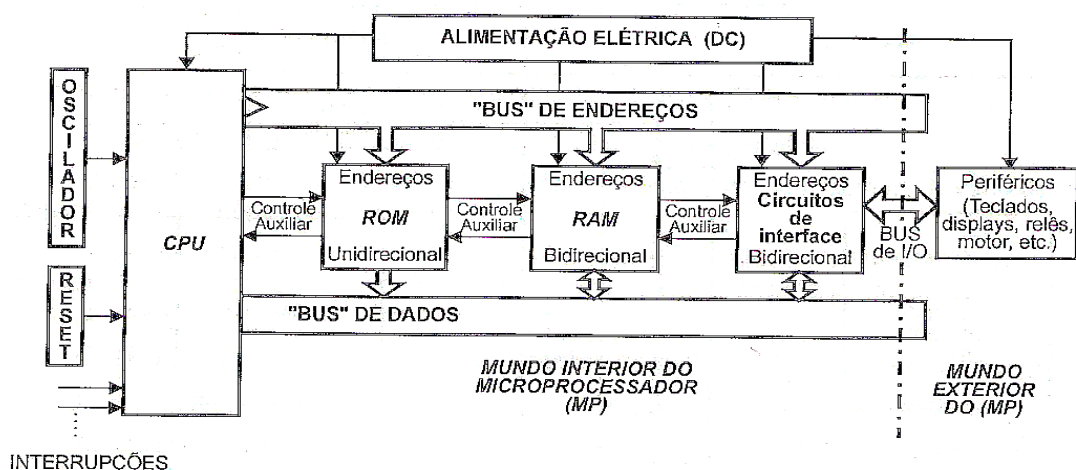


Figura 2.6 – arquitetura básica do microprocessador. (NICOLSI, 2000, p.60)

- BUS: barramento de dados e barramento de endereço.
- Controle de endereço: serve para o MP selecionar com qual posição de memória ou periférico deseja se comunicar.
- Controle auxiliar: são sinais de controle que permite o MP acionar em determinado tempo a memória ROM e não a memória RAM ou vice-versa.
- “I/O” (*Input/ Output* ou Entrada/Saída): serve de comunicação com os elementos eletroeletrônicos.
- CPU: é o componente principal do sistema.

A CPU trabalha somente com números binários, já o programador entende números octal, hexadecimal, decimal e nomes simbólicos.

A CPU possui algumas funções internas como:

- Registradores: tem a função de gravar dados temporários internamente e externamente da CPU, semelhante à memória RAM.
- Controlador de programa: endereço de memória externa.
- Registrador de instrução: código de instruções que são registradas na ROM por meio de endereço do PC no ciclo de busca da instrução da memória.

- Unidade decodificadora: é a unidade responsável para decodificar as instruções.
- Unidade lógica e aritmética (ALU): é responsável para realizar todas as operações lógicas, aritméticas e decisão/comparação de uma máquina.
- Acumulador: é um registrador principal.
- Unidade de controle: área responsável por processar o controle do fluxo das informações a fim de realizar a instrução recebida.

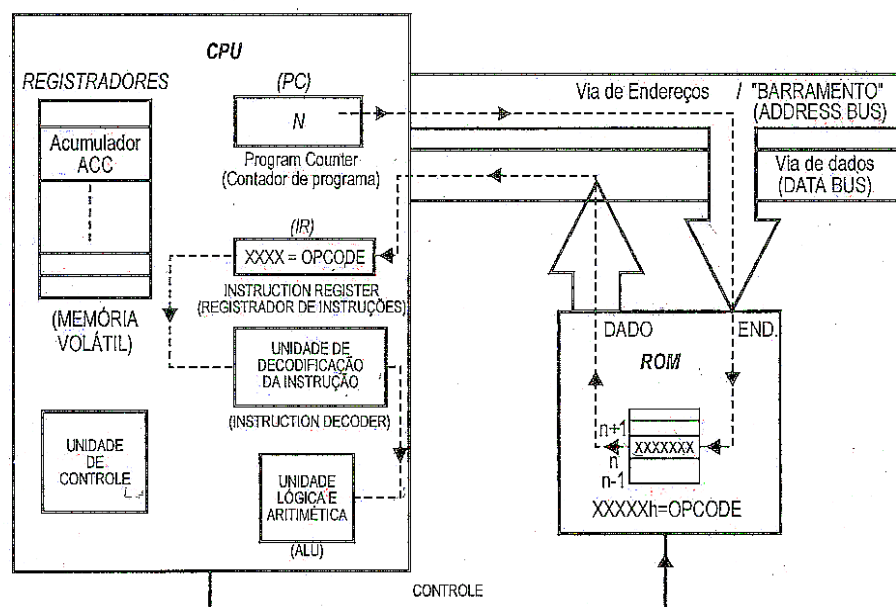


Figura 2.7 – Ilustração da CPU em ciclo de busca de instrução (na ROM). (NICOLSI, 2000, p.62).

A figura 2.7 ilustra o funcionamento de uma CPU buscando os dados em uma memória ROM, da seguinte forma: o endereço da instrução (N) a ser lido é carregado no controlador de programas (PC), que percorre a via de endereço, onde o sinal de controle da ROM é ativado. Este sinal da ROM registra novos dados (N+1), que percorrem um novo caminho através da via de dados, em que as instruções são carregadas e armazenadas no registrador de instruções (IR). A execução destes dados é interna na CPU, auxiliada pela unidade de controle e pelos registradores, decodificado pela unidade de decodificação da instrução e executado pela ALU.

2.6 Microcontrolador 8051 e as especificações do AT89S8252

2.6.1 Histórico do microcontrolador

O primeiro microcontrolador foi lançado pela Intel em 1978 e recebeu a sigla de 8048, que trabalha com apenas palavras de 4 bits. Logo depois, evoluiu e recebeu uma nova sigla, utilizada até os dias atuais, que é a da família 8051, surgida em 1983. A Intel oferece, ainda, o microcontrolador da família 8096, que trabalha com 16 bits, possibilitando maior capacidade de processamento.

O microcontrolador pode ser considerado como computador programável, em um *chip* otimizado, para controlar dispositivos eletrônicos. É uma espécie de microprocessador, com memória e *interfaces* de E/S (Entrada/Saída) integrados. O *chip* é um circuito interligado, funcionando como um dispositivo microeletrônico, que consiste em muitos transistores e outros componentes interligados, capazes de desempenhar várias funções.

Depois que surgiram os microcontroladores da *INTEL*, este produto se espalhou rapidamente, sendo que hoje há uma grande variedade de produtos para solucionar diferentes problemas de controle, tais como o da *Zilog*, com sua família Z8; a *National*, com o COP8; a Motorola, com o 6811 e a *Microchip*, com seus *PICs*.

No entanto, nenhum desses microcontroladores ficou tão conhecido como o da família 8051. Assim, este microcontrolador foi produzido por outras empresas, que implementaram inovações. Nos dias atuais, o referido microcontrolador se destaca como pertencendo à família que oferece maior variedade de opções.

2.6.2 Diferença entre Microcontrolador (MC) e Microprocessador (MP).

A figura 2.8 mostra a diferença entre o microcontrolador (MC) e um microprocessador (MP).

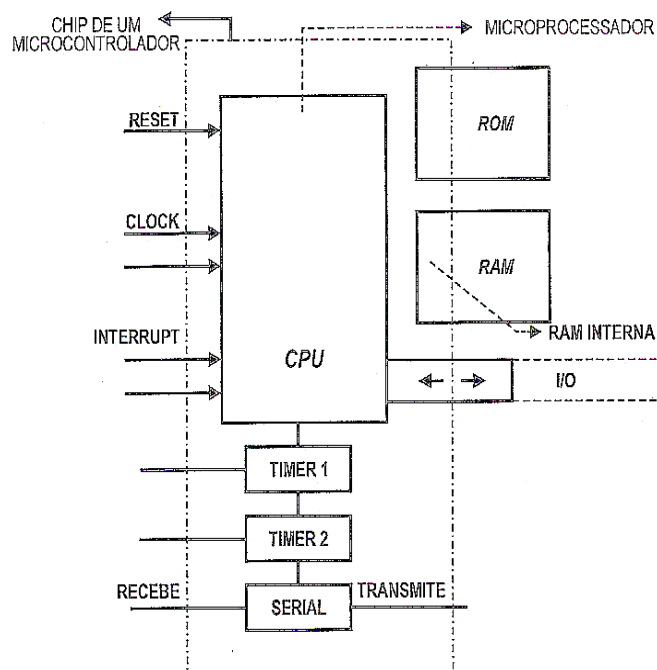


Figura 2.8 - Diferença entre MC e MP. (NICOLOSI, 2000, p.65).

A diferença entre o MC e MP se dá através do hardware interno. O microcontrolador possui muito mais funções que o microprocessador.

Os microprocessadores internamente possuem além do chip do microprocessador, da ROM, do “Latch” e da RAM, outros chips auxiliares, como “Timers” (que contam o tempo) e “Serial”. O microcontrolador é idêntico ao microprocessador, só que ele possui todos os periféricos do microprocessador em um mesmo chip.

2.6.3 Mundo Externo – Entrada/ Saída (I/O) e Periféricos

O microcontrolador deve ter suas funções voltadas para controlar equipamentos de forma a facilitar o cotidiano do seres humanos, como o controle de impressoras, teclados, acionamento de lâmpadas entre outros periféricos que necessitam de controle.

Divide-se basicamente em:

1) Periféricos de Armazenamento de Massa – Um Cd-rom, um HD (Hard Disc), um DVD, uma memória RAM, memória ROM, entre outros, que são tipos de periféricos de saída de dados.

2) Interface Homem-Máquina – é a comunicação do homem com a máquina, como por exemplo: display, teclado, mouse, joystick, vídeo, pen drives, entre outros. Estes tipos de periféricos são de Entradas de dados.

3) Periféricos de Controle: Atuação e Sensoriamentos – O microcontrolador “enxerga” um dispositivo externo e consegue resolver com outro periférico externo. Por exemplo, o “A/D” (Conversor Analógico-Digital) comunica-se com microcontrolador, transformando suas variáveis de saída em variáveis de entradas; o “D/A” (Conversor Digital-Analógico) tem a função de fazer com que a saída do microcontrolador passe através de um sinal analógico, antes de controlar um dispositivo (dispositivo de saída, como relês, lâmpada, entre outros).

A figura 2.9 mostra o microcontrolador e seus periféricos de entrada e saída:

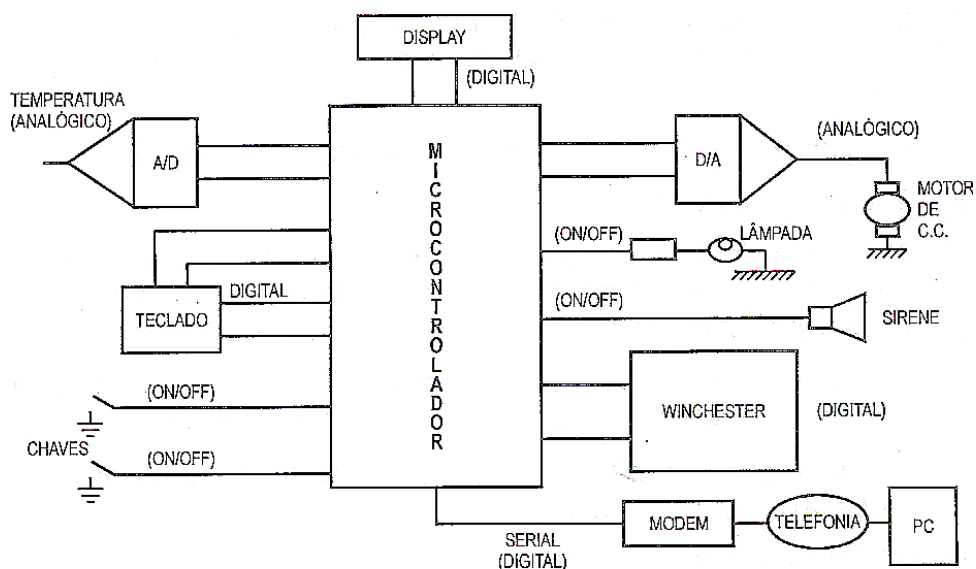


Figura 2.9 – Microcontrolador e seus periféricos de entrada e saída. (NICOLSI, 2000, p.67).

2.6.4 Microcontrolador 8051

A arquitetura do microcontrolador da família 8051 pretende que os recursos específicos devam ser compatíveis com ele. O interessante na arquitetura

dos 8051, ao contrário do computador, é que ela separa a memória de programa da memória de dados. As memórias são organizadas em blocos para oferecer maior versatilidade às aplicações.

A figura 2.10 mostra o microcontrolador da família 8051.

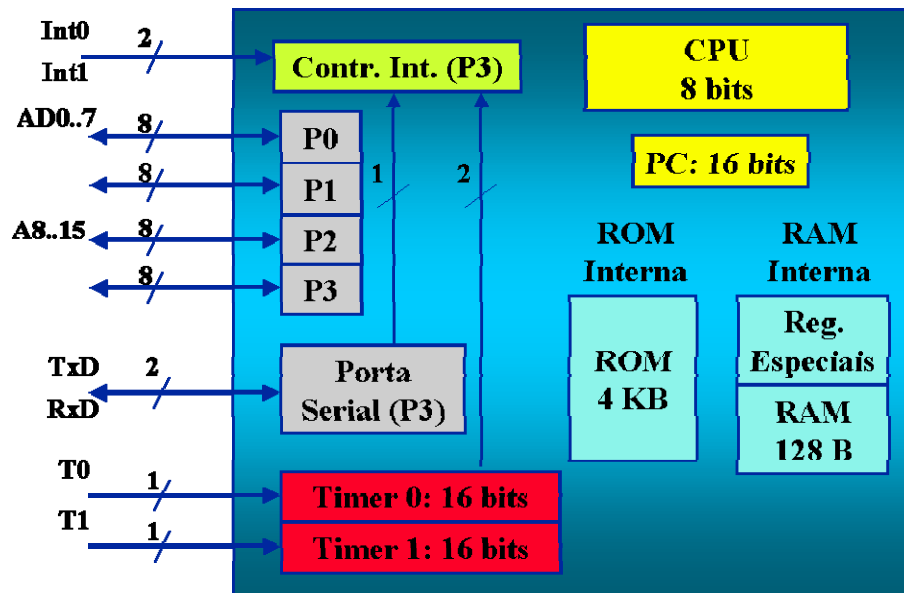


Figura 2.10 – Funcionamento do microcontrolador 8051.

(http://www.mzeditora.com.br/artigos/mic_modernos.htm)

A memória Flash tem a função de armazenar dados por longos períodos, sem precisar de alimentação elétrica.

A memória Flash pode ter 64 KB, destinada a programa, e 4 KB, para dados. A gravação da memória Flash pode ser através de portas seriais.

A rápida evolução da microeletrônica levou a um grande aceleração de microprocessadores, com isso surgiu o conceito da computação invisível.

Computação Invisível ou Ubíqua é a tecnologia imperceptível, ou seja, a comunicação entre computadores ou eletrônicos de forma automática sem a necessidade dos seres humanos informarem dados aos computadores ou eletrônicos.

A figura 2.11 mostra a interligação básica do microcontrolador 8051.

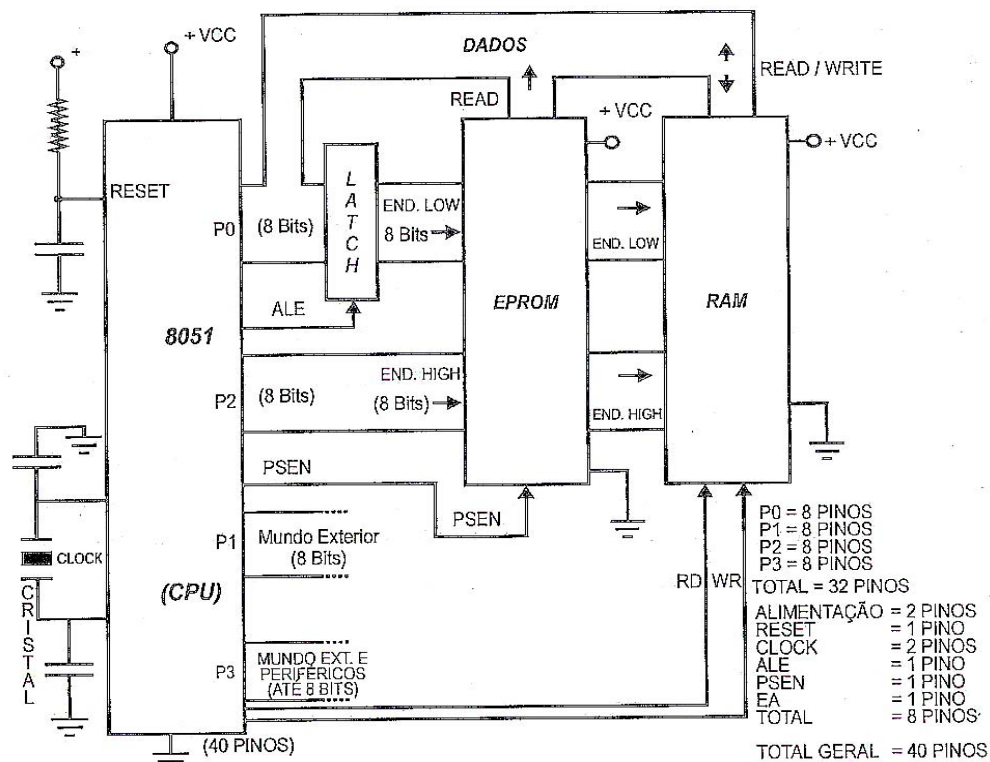


Figura 2.11 – Diagrama de interligação básica do microcontrolador 8051. (NICOLSI, 2000, p.64).

O barramento de endereço é composto de 16 *bits*, sendo separado dois de 8 *bits*, nos quais 8 *bits* menos significativos são derivados juntamente com os 8 *bits* de dados. O “Latch” é responsável por separar os dados do endereço, que é copiado, quando este é apresentado por P0. Isto se faz por meio de pino de comando chamado “ALE”. Os dados e endereços são multiplexados pela via P0.

O microcontrolador da família 8051 possui 40 pinos.

O pino P2 possui os outros 8 bits mais significativos do barramento de endereço, que são diretos, e não multiplexado como a P0.

O PSEN é um pino de acionamento da *ROM* (conhecido como *EPROM*).

Os pinos de “Read” ou “Write” são pinos de acionamento da RAM.

Os pinos “P1” e “P3” são portas destinadas à interface externa de “leitura” e “escrita”, no mundo externo, ou seja, ler um teclado, escrever em display.

O 8051 é um microcontrolador que trabalha com palavra de 8 bits, possui alta performance e custo baixo na implementação de circuitos eletrônicos. Para que se possa programar este microcontrolador deve-se conhecer as configurações internas deste hardware:

- Entrada de interrupções externa;
- Instrução interna de multiplicação e divisão;
- Ciclos típicos de instrução de 1 e 2 nano microsegundos a 12 MHz.
- Processador “Booleano” (bits);
- Capacidade de 64 Kbytes de endereçamento externo de RAM;
- Capacidade de 64 Kbytes de endereçamento externo de ROM;
- 1 Interface serial *full duplex* programável (UART).
- 3 Timers de 16 bits;
- 4 Portas de I/O(Entrada/Saída): P0, P1, P2 e P3;
- Cada Porta de I/O possui 8 *bits*.
- ROM interna de 4 Kbytes;
- RAM interna de uso geral de 128 bytes e 128 bytes correspondentes aos registradores especiais.
- 9 possibilidades de interrupções, incluído o *reset*, com dois grupos de prioridades.

2.6.5 Arquitetura Interna dos 8051

A figura 2.12 mostra a arquitetura interna do microcontrolador 8051.

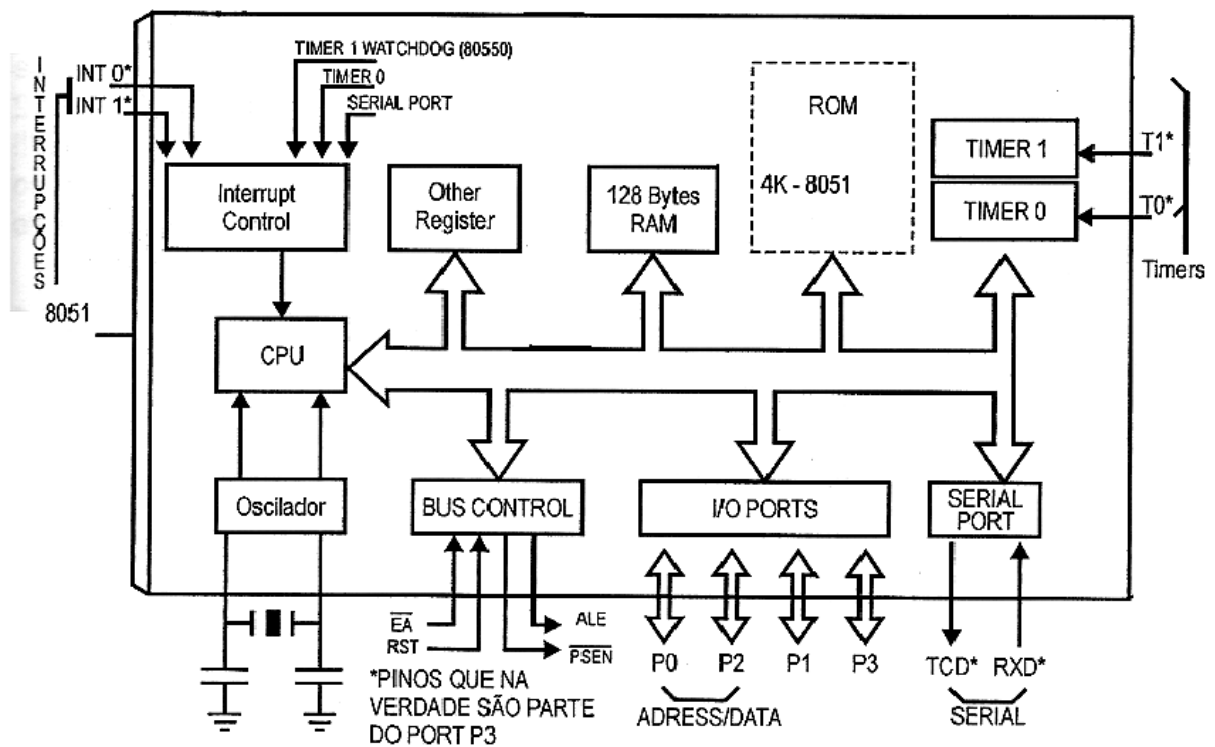


Figura 2.12 – Diagrama de bloco funcional do MC 8051. (NICOLOSI, 2000, p.71)

A comunicação externa do microcontrolador é feita pelas portas P0, P1, P2 e P3, cada uma com 8 linhas *bits*. Cada porta tem uma função. Por exemplo, as portas P0 e P2 servem para gerenciar as vias de dados e endereços da comunicação de microcontrolador com a *RAM*, *ROM* ou periféricos de “I/O mapeados”. As portas P1 e P3 servem para comunicação com mundo externo.

A porta P3 possui uma comunicação especial, pois existem 2 *timers*, 1 *serial* e A/D, se utilizarmos a porta P1 que é totalmente livre.

A tabela 2.4 mostra a comparação de recursos do MC.

Código	ROM	RAM	A/D	Timers	Serial	Utilização das portas P3 e P1															
						P1								P3							
						0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
8051	4K	128	Não Tem	2	1	P 1 0	P 1 1	P 1 2	P 1 3	P 1 4	P 1 5	P 1 6	P 1 7	R X	T X	I N T 0	I N T 1	T 1	T 0	W R	R D
8031	Não Tem	128	Não Tem	2	1	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
8751	4K - EPROM	128	Não Tem	2	1	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
8052	8K	256	Não Tem	3	1	T 2	T 2 E X	*	*	*	*	*	*	*	*	*	*	*	*	*	*
8032	Não Tem	256	Não Tem	3	1	T 2	T 2 E X	*	*	*	*	*	*	*	*	*	*	*	*	*	*
8752	8K - EPROM	256	Não Tem	3	1	T 2	T 2 E X	*	*	*	*	*	*	*	*	*	*	*	*	*	*
80550	Não tem	128	8 Canais	2	1	A / D	A / D	A / D	A / D	A / D	A / D	A / D	A / D	*	*	*	*	*	*	*	*
83550	4K	128	8 Canais	2	1	A / D	A / D	A / D	A / D	A / D	A / D	A / D	A / D	*	*	*	*	*	*	*	*
85550	4K - EPROM	128	8 Canais	2	1	A / D	A / D	A / D	A / D	A / D	A / D	A / D	A / D	*	*	*	*	*	*	*	*

Tabela 2.4 – Comparação dos recursos dos MC citados.(NICOLSI, 2000, p.73)

2.6.6 Descrição Formal da Pinagem do MC 8051

A figura 2.13 mostra o esquema elétrico do microcontrolador 8051.

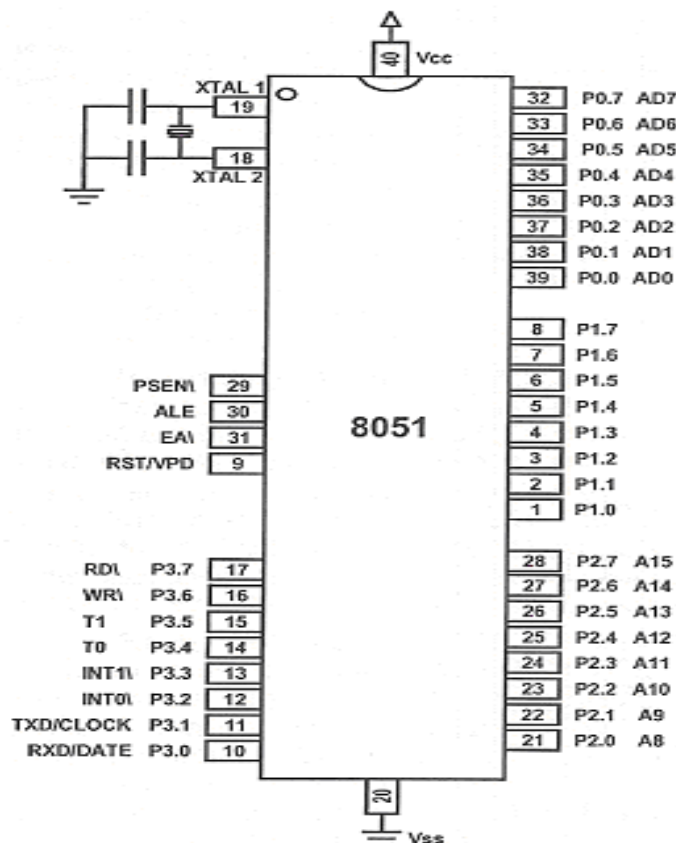


Figura 2.13 – Chip do 8051 (NICOLSI, 2000, p.74).

- Port P0 - Na Porta P0 os AD1 até AD7. A palavra “AD” significa “Address/Data”. Portanto, a porta P0 é um “multiplexador” entre as funções de enviar endereços (*address*) e dados (*data*) pelo mesmo caminho chamado “P0”.

Esta porta quando não utilizada para leitura e escrita de memória externa, bem como dados para *Flash*, utiliza-se para interligar dispositivos. A P0 possui a característica de “dreno aberto”, ou seja, esta porta possui uma baixa impedância (0 volts), com isso necessita de utilizar resistores para que seja feito *pull-up* externos a fim de garantir uma alta impedância (5 volts) e um correto

funcionamento entre o microcontrolador e os dispositivos. As portas do P0 no AT89S8252 apresentam dreno aberto, os demais apresentam resistores de *pull-up* internos. A diferença entre uma porta com “dreno aberto” ou *pull-up* externo e uma porta com *pull-up* interno está na figura 2.14.

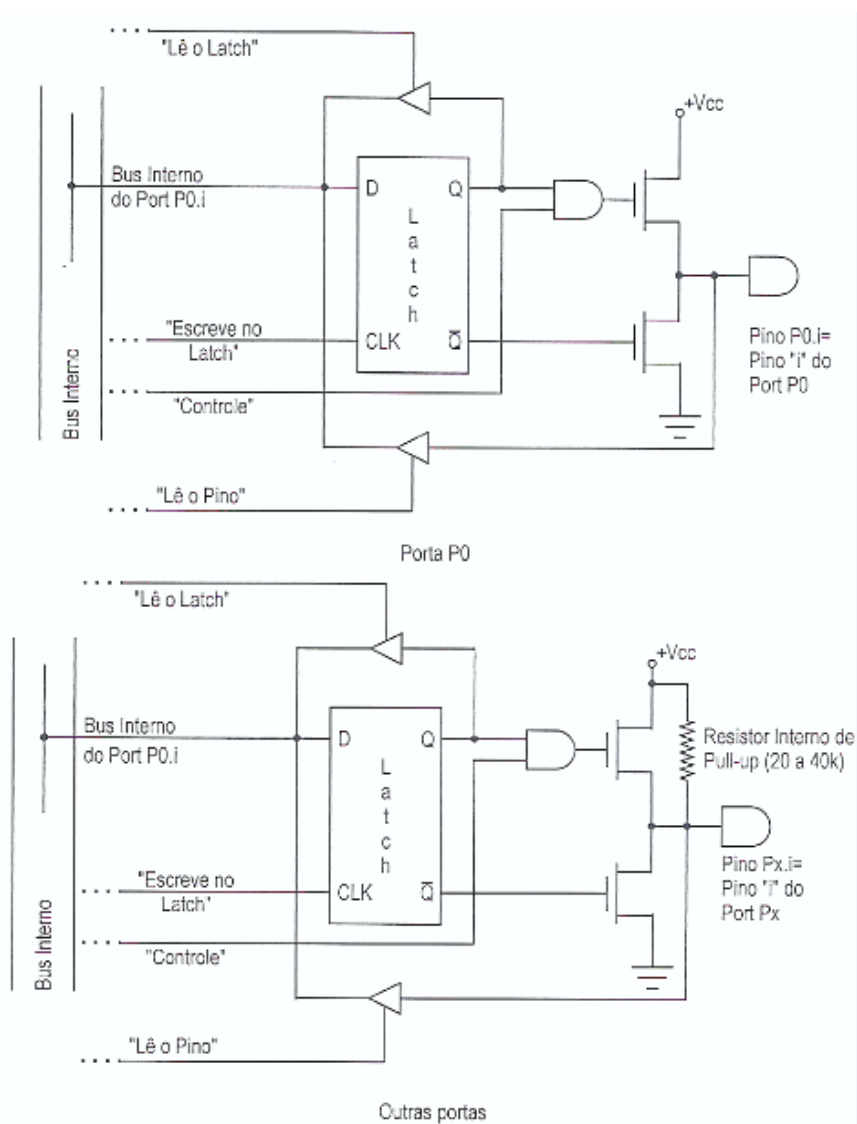


Figura 2.14 – Representação das portas. (NICOLOSI, 2005, p.27).

Os dois tipos de configuração possuem uma diferença de fornecer correntes as portas que possuem resistência interna, ou seja, ao forçar a ida da corrente para o terra, tem se o nível lógico zero para essa porta.

- Port P1 – Na Porta P1 utilizam-se oito *bits* de propósito geral com “I/O”, possuem resistores de *pull-up* internos. Esses resistores, ao serem colocados em nível lógico 1, as portas terão a capacidade de gerar corrente(I_{IL}) com a interface de outros dispositivos.

As portas do P1 possuem funções que podem ser configuradas pelo programador, como a P1.0 e P1.1 responsável pelo terceiro *timer*, enquanto o P1.4 a P1.7 estão destinados ao sistema SPI de gravação. A tabela 6 mostra as funções atreladas a cada porta do *port* P1.

Tabela 2.5 - Funções atreladas as portas da porta P1.

Portas	Função
P1.0	T2 – Entrada externa para uso no modo contador para <i>timer/counter 2 – Clock out</i>
P1.1	T2EX – Captura do sinal para gerar interrupção do T2
P1.2	-
P1.3	-
P1.4	SS – Porta de seleção de entrada
P1.5	MOSI – <i>Master Data input</i> e <i>Slave Data output</i> para SPI
P1.6	MISO – <i>Master Data output</i> e <i>Slave Data output</i> para SPI
P1.7	SCK – <i>Master Clock output</i> , <i>Slave Clock input</i> para SPI

Tabela 2.5 – Funções da porta P1.(NICOLOSI, 2005, p.28)

- Port P2 – Na porta P2 utilizam-se para comunicação de oito portas de vias de endereço e com resistores de *pull-up* internos. O propósito geral da *port* é utiliza a memória RAM/EPROM/ROM externa, sendo a parte mais significativa dos endereços, ou como I/O normal. Esta port também tem destino de receber bits mais significativos e alguns sinais de controle durante a programação paralela e a verificação da memória *flash*.

- Port P3 – A porta P3 possui 8 *bits*, além de resistores com *pull-ups* internos. As portas tem como propósito geral o fato de que se as mesmas

não forem utilizadas por nenhum periférico interno ou pela *RAM* externa serão utilizados para recurso de I/O. Figura 2.15 ilustração de utilização da porta P3.

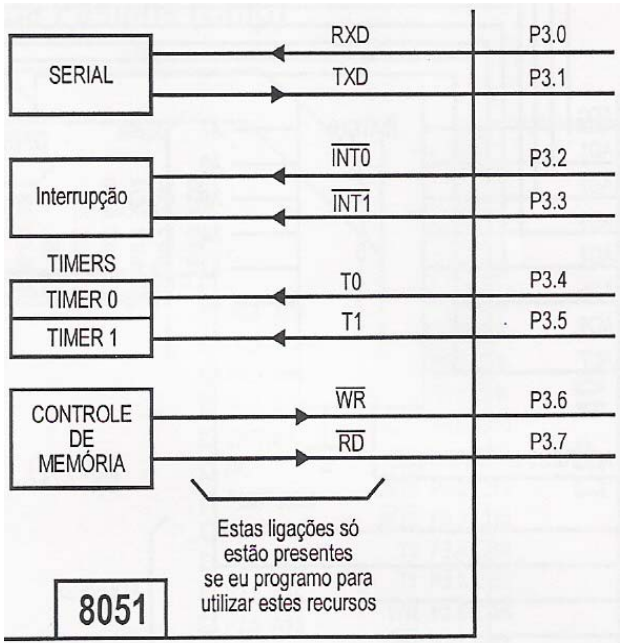


Figura 2.15 – Ilustração da porta P3 (NICOLSI, 2000, p.78)

Observação: Se não for utilizado nenhum destes recursos, o pino correspondente ao recurso não utilizado será livre para I/O de uso geral.

A tabela 2.6 resume a ilustração anterior.





Nome	Número do Pino	Função Especial	Função Normal	Função Especial	Comentário da Função Especial
P3.0	10	RXD	I/O	RD _X , Recebimento de Dados	Usado como receptor de dados serial
P3.1	11	TXD	I/O	TXD, Transmissão de Dados	Usado como transmissor de dados serial
P3.2	12	 INT0	I/O	Interrupção Externa 0	Usado para algum evento externo interromper o MC.
P3.3	13	 INIT 1	I/O	Interrupção Externa 1	Usado para outro evento externo interromper o MC.
P3.4	14	T0	I/O	Timer/Counters 0:Entrada Externa	Usado quando se quer que um timer zero se torne um contador de eventos externos.
P3.5	15	T1	I/O	Timer/Counters 1:Entrada Externa	Usado quando se quer que um timer 1 se torne um contador de eventos externos.
P3.6	16	 WR	I/O	Dados Externos: Memória somente de leitura	Usado quando se conecta a RAM externa no chip. Sinaliza que o Mp vai “escrever” na RAM.
P3.7	17	 RD	I/O	Dados Externos: Memória somente de escrita	Usado quando se conecta a RAM externa no chip. Sinaliza que o Mp vai “ler” na RAM.

Tabela 2.6 – Resumo das funções especiais da porta P3. (NICOLSI, 2004, p.78)

- **PSEN (*Program Store Enable*):** é um pino de controle da *ROM/RAM*. O Mc faz uma busca na *ROM*, para em seguida, executá-la. O traço acima da palavra significa que ele é ativo em nível lógico 0.
- **ALE (*Address Latch Enable*):** é o pino de controle de comunicação de P0 com a via de dados e endereço e acesso a memória externa. Responsável por demultiplexar dados e os bits menos significativos dos endereços de acessos.
- **EA (*External Access*):** Tem a função de mapear a *ROM* externa ou interna. Se o nível lógico for 1, o *chip* irá ler sua *ROM/EPROM* interna. E se o nível lógico for 0, o chip irá ler sua *ROM/EPROM* externa.

- RST (*Reset*): é o pino de inicialização adequada da máquina. O nível lógico tem que estar sempre em 1 pelo menos dois ciclos da máquina para saber o futuro. O RST normalmente utiliza um capacitor e um resistor para produzir o efeito, tornando-se automático.
- “XTAL 1 e XTAL 2 (Cristal/Oscilador) – Esse *chip* tem um sistema de oscilação interna que só exige do exterior, o cristal e dois capacitores para gerar a oscilação, que se tornará o clock ou padrão de tempo para o microcontrolador trabalhar. “(NICOLSI, 2004, p.80)
- “Vcc e Vss (Alimentação do *chip*) – Alimentação de energia do chip: +5 Vdc em Vcc, pino 40 e terra em Vss, pino 20.”

2.6.7 Organização das memórias.

A figura 2.16 mostra os diferentes tipos de memórias da família 8051.



Figura 2.16 – Diferentes tipos de memórias do 8051. (NICOLSI, 2005, p.21).

2.6.7.1 Memória para Código (CODE)

Nessa região de memória se encontra o programa executável. No microcontrolador 80C51, da família do 8051, encontram-se 4 *Kbytes* de memória

ROM para essa finalidade. No AT89S8252, este microcontrolador possui um tipo de memória capaz de ser reprogramada eletricamente. Este tem capacidade de fazer o *download* de até 8 Kbytes.

2.6.7.2 Espaço para Manipulação de Dados (DATA)

Essa região de memória destina-se à manipulação de dados. Para que esta manipulação ocorra utiliza-se uma memória do tipo RAM.

A tabela 2.7 explora a região da memória DATA.

Intervalo de Endereço (HEX)	Nome	Observações	Região
7F ↑ 30	Byte endereçável	Posição de memória acessível apenas por <i>byte</i>	Região C
2F ↑ 20	Byte endereçável	Posição de memória acessível por <i>bit</i> e por <i>byte</i>	Região A
1F ↑ 18	Banco 3	Registradores em grupo de 4 bancos	Região A
17 ↑ 10	Banco 2	Endereçamento direto	Região A
0F ↑ 08	Banco 1	Endereçamento Indireto	Região A

<p>07 ↑ 00</p>	Banco 0	Endereçamento pelo <i>Stack Pointer</i>	Região A
------------------------	---------	--	----------

Tabela 2.7 – Região *DATA* (NICOLOSI, 2005, p.22).

Na tabela 2.8 encontram-se três regiões. A primeira região dividi-se em quatro grupos, e cada um possui oito registradores de 8 *bits* que vão de R0 a R7. A região B e C se diferenciam pelo modo de acesso aos *bits* dos registradores. O bloco C possui acesso somente aos *bytes* e o bloco B, além dos acessos aos *bytes* pode também acessar os *bits*.

2.6.7.3 Registradores para Funções Especiais (*Special Function Register – S.F.R*)

Os registradores especiais estão alocados em uma memória RAM internos que tem início no endereço 80h e término no endereço FFh. Cada um destes registradores especiais são compostos de 8 *bits*, onde na maioria dos casos cada bit possui uma função especial, que esta vinculada a alguma função do microcontrolador.

2.6.7.4 Extensão da Memória RAM (*IDATA*)

O microcontrolador AT89S8252, da família 8051, apresenta maior quantidade de memória RAM interna, proporcionada pela região *IDATA*. Para que não haja conflitos nas áreas destinadas aos registradores especiais, os dados deveram ser referenciados somente de forma indireta, já que ambas as regiões de memórias possuem os mesmos endereços. Portanto, o acesso a região *IDATA* será mais demorado do que à região *DATA*, justamente pela quantidade maior de números de instruções para leitura e escrita.

2.6.7.5 Extensão de Memória da Dados Externa (XData)

A *XDATA* foi intitulada como memória externa, por sua implementação ser equivalente à de um periférico. Essa região normalmente utiliza uma memória *SRAM* (*RAM* estática). O acesso a extensão de memória de dados externos pode ser feito pelo *DPTR* (*Data Pointer Register*) ou através de registradores de forma indireta.

2.6.7.6 RAM Interna

O microcontrolador da família 8051 possui um espaço de 256 bytes, que é dividido em:

1. Posições da *RAM* “com apelidos”: As posições da memória podem ser acessadas através do endereço absoluto, que são divididos em quatro bancos e cada um tem seu endereço: R0 a R7. Os registros de memória vão de 00h até 1Fh, dados da tabela ASCII, de endereço inteiro.
2. Posições da *RAM* “sem apelidos”: São acessadas somente através do endereço absoluto, mas com uma propriedade adicional: cada bit de uma posição de memória tem seu endereço particular, isto é, ela é *bit* endereçável, além de ser *byte* endereçável. As posições de *bytes* vão de 20h até 2Fh.
3. Registradores de funções especiais: são aqueles que além de terem posições da *RAM* com apelidos, e seu endereço absoluto, tem funções específicas no microcontrolador.

BYTE ADDRESS		BYTE ADDRESS								
"NOT BIT ADDRESSABLE"		7F	GENERAL PURPOSE RAM							
		30								
BIT ADDRESSABLE LOCATION	2F	7F	7E	7D	7C	7B	7A	79	78	
	2E	77	76	75	74	73	72	71	70	
	2D	6F	6E	6D	6C	6B	6A	69	68	
	2C	67	66	65	64	63	62	61	60	
	2B	5F	5E	5D	5C	5B	5A	59	58	
	2A	57	56	55	54	53	52	51	50	
	29	4F	4E	4D	4C	4B	4A	49	48	
	28	47	46	45	44	43	42	41	40	
	27	3F	3E	3D	3C	3B	3A	39	38	
	26	37	36	35	34	33	32	31	30	
	25	2F	2E	2D	2C	2B	2A	29	28	
	24	27	26	25	24	23	22	21	20	
	23	1F	1E	1D	1C	1B	1A	19	18	
	22	17	16	15	14	13	12	11	10	
	21	0F	0E	0D	0C	0B	0A	09	08	
	20	07	06	05	04	03	02	01	00	
NOT BIT ADDRESSABLE	1F	BANK 3								
	18	BANK 3								
	17	BANK 2								
	10	BANK 2								
	0F	BANK 1								
	08	BANK 1								
	07	DEFAULT REGISTER BANK FOR R0-R7								
	00									

RAM

BYTE ADDRESS		BYTE ADDRESS								
FF		F7	F6	F5	F4	F3	F2	F1	F0	B
F0	E7	E6	E5	E4	E3	E2	E1	E0		ACC
D0	D7	D6	D5	D4	D3	D2	D1	D0		PSW
B8	-	-	-	BC	BB	BA	B9	B8		IP
B0	B7	B6	B5	B4	B3	B2	B1	B0		P3
A8	AF	-	-	AC	AB	AA	A9	A8		IE
A0	A7	A6	A5	A4	A3	A2	A1	A0		P2
99	NOT BIT ADDRESSABLE									SBUF
98	9F	9E	9D	9C	9B	9A	99	98		SCON
90	97	96	95	94	93	92	91	90		P1
8D	NOT BIT ADDRESSABLE									TH1
8C	NOT BIT ADDRESSABLE									TH0
8B	NOT BIT ADDRESSABLE									TL1
8A	NOT BIT ADDRESSABLE									TL0
89	NOT BIT ADDRESSABLE									TMOD
88	8F	8E	8D	8C	8B	8A	89	88		TCON
87	NOT BIT ADDRESSABLE									PCON
83	NOT BIT ADDRESSABLE									DPH
82	NOT BIT ADDRESSABLE									DPL
81	NOT BIT ADDRESSABLE									SP
80	87	86	85	84	83	82	81	80		P0

SPECIAL FUNCTION REGISTER

Figura 2.17 – Memória RAM.(NICOLOSI, 2000, p.83).

- ACC – Acumulador – Utilizado para muitas instruções de máquinas.
- P0, P1, P2 e P3 – Registradores de portas de comunicação do microcontrolador.
- B – Registrador utilizado para multiplicação e divisão.
- PSW – Registrador do estado de ultima operação lógica e aritmética realizado no acumulador.
- IP e IE – Registradores utilizados para programar interrupções no microcontrolador.

- DPH e DPL – Ambos juntos formam um registrador de 16 bits chamado de *DPTR*, que permite criar um endereço de acesso às memórias externas.

- SP – São registradores utilizados como pilha de endereço de retorno para sub-rotinas, ou seja, serve para que o microcontrolador saiba o caminho de volta depois de uma interrupção programada.

- PCON – Usado para alterar modos de funcionamento da pastilha com relação ao consumo de potência.

Registradores que correspondem a periféricos internos que fazem comunicação com microcontrolador.

- SBUF – Chip de comunicação. Faz comunicação interna e também comunicação externa, quando se quer transmitir/receber dados via serial.

- SCON – Permite programar e controlar todas as atividades dos periféricos seriais juntamente com SBUF.

- TH0, TL0, TH1 e TL1 – São registradores de 8 bits, mas em conjunto de dois (high e low) formam um timer (timer 0 e timer 1) registrador de contagem de 16 bits.

- TMOD – Permite programar cada timer.

2.6.7.7 Registradores Especiais Adicionais Presentes no AT89S8252

- DP0L e DP0H / DP1L e DP1H – O AT89S8252 possui dois *DPTR* (*Data Pointer*) de 16 bits para simplificar o acesso às memórias. O registrador WMCON faz a seleção através do *bit* especial DPS (*Data Pointer Register Select*).

- “SPDR – Região de memória composta por 8 bits.Registrador de dados para o SPI.”(NICOLOSI,2005,P.33).

- WMCON – Esse registrador permite além do controle do *Watchdog timer*, o acesso aos bits especiais da EEPROM, conhecidos como

EEMEN e EEMWE. Além deste controle possuem também o *bit* DPS para selecionar o DPTR0 e DPTR1.

- T2COM – Controla os registradores do *Timer2*.
- T2MOD – Seleção do modo de operação do registrador *Timer2*.
- “SPCR –SPI *Control Register* – Registrador de controle para o sistema SPI.” (NICOLSI,2005,P.33).
- “SPSR – SPI *Status Register* – Registrador que contém o *status* da comunicação SPI.” (NICOLSI,2005,P.33).
- RCAP2L e RCAP2H – Funções dos registradores do *Timer 2* que variam conforme o modo de trabalho.
- TL2 e TH2 – Registradores vinculados ao *Timer 2* que juntos tem a função de fornecer o valor para contagem.

2.6.8 Clock e Reset

2.6.8.1 Clock

“É um relógio do microcontrolador para execução seqüencial de qualquer atividade interna ou externa à máquina”. (NICOLSI,2005,P.33).

O ciclo de máquina se dá através de um conjunto de seis estados, os estados são formados por dois pulsos e os pulsos são formados através de cada ciclo de oscilação que são conhecidos como *clock*.

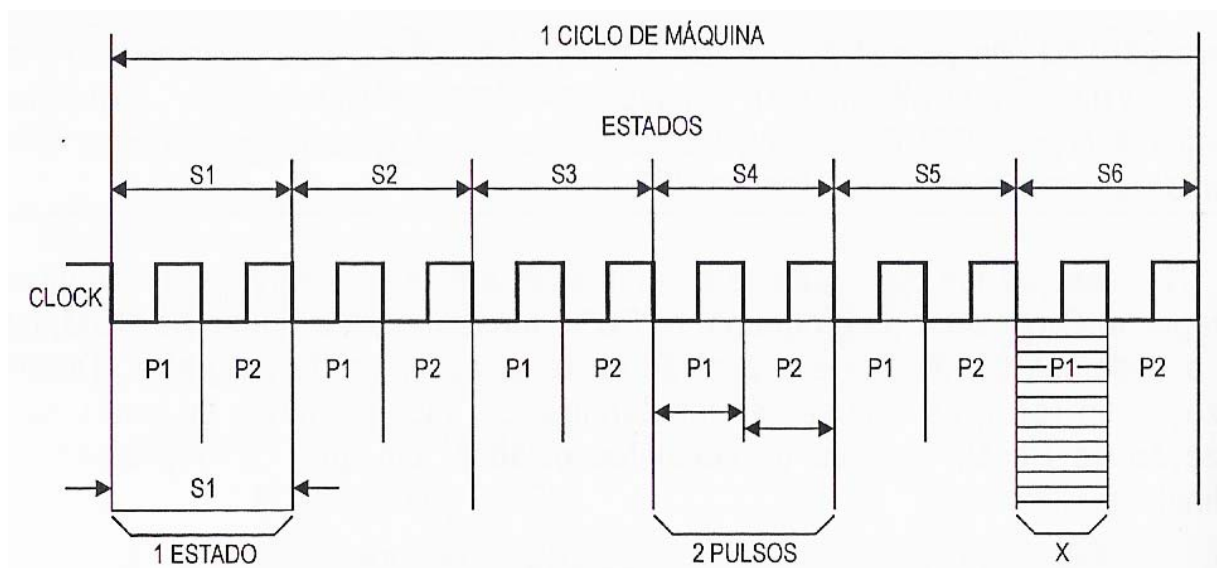


Figura 2.18 – Ilustração de um ciclo de máquina no MC 8051. (NICOLSI, 2000, p.99)

2.6.8.2 **Reset**

O *reset* é um pino físico no microcontrolador. Para que este pino funcione é necessário um circuito esteja conectado a este pino, para que no acionamento do chip, o *reset* fique pelo menos dois ciclos de máquinas no estado "1" ou "*high*". A função do *reset* deve ser programada automaticamente na energização do circuito e também para que seja forçado por uma chave.

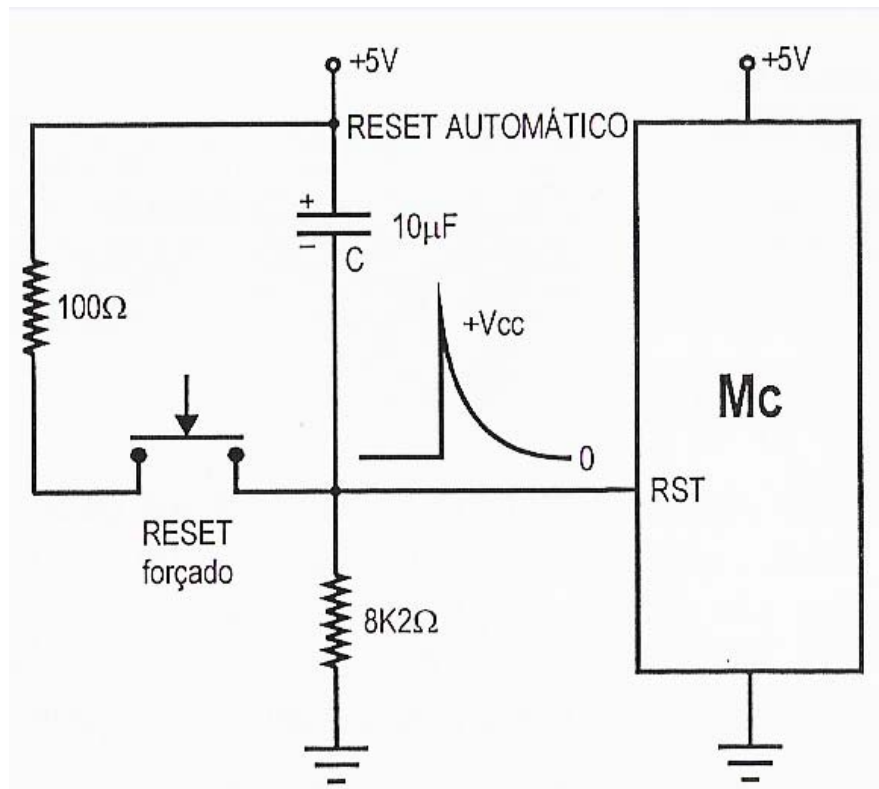


Figura 2.19 – Reset automático e o Reset forçado. (NICOLSI, 2000, p.100).

2.7 Registradores Especiais.

Existem alguns registradores que não permitem acessar seus bits especiais. Quando não se trabalha com esse tipo de registradores, se deve atribuir ao registrador um valor que compreenda os 8 *bits* especiais.

Os registradores de *bits* especiais são divididos em dois grupos distintos:

- Primeiro Grupo – Registradores que permitem acesso direto aos bits especiais: SCON, IE, TCON, IP, T2COM, PSW, B, A.
- Segundo Grupo – Registradores que não permitem acesso direto aos bits especiais: TCON, TMOD, T2MOD, WMCON, SPCR, SPSR, SPDR. Este segundo grupo se divide em três grandes grupos de registradores: os vinculados às interrupções, os vinculados aos *timers* e os vinculados à comunicação serial.

2.7.1 Interrupções

É um evento interno ou externo em que o microcontrolador é obrigado a interromper suas atividades em execuções, para atender o evento que causou interrupção.

Após a interrupção o microcontrolador volta a realizar a atividade anterior, partindo do ponto em que se encontrava.

A figura 2.20 mostra a ilustração do processo de interrupção.

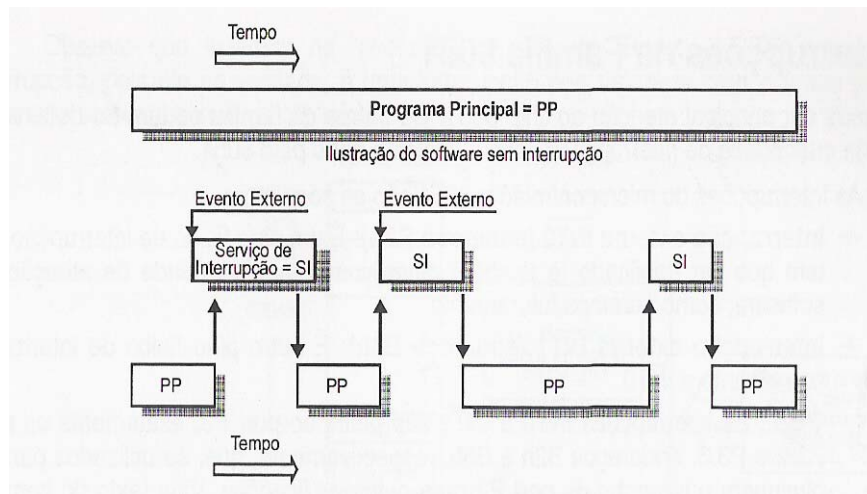


Figura 2.20 – Ilustração do processo de interrupção. (NICOLOSI, 2000, p.145).

2.7.1.1 Propriedades da interrupção

a) Vetorada ou não vetorada: O microcontrolador a ser interrompido deve ir para o endereço de interrupção solicitada. Interrupção não vetorada é quando o microcontrolador tem um endereço fixo de desvio de interrupção. Interrupção vetorada é quando o microcontrolador permite a interrupção e o dispositivo que o interrompeu envia o endereço de desvio da interrupção.

b) Mascaramento: O microcontrolador tem uma propriedade que permite via software ou hardware permitir ou não a interrupção de certos dispositivos.

c) Prioridade: No microcontrolador deve se programar a prioridade de determinados dispositivos, para que não haja conflito na interrupção.

d) Origem: A interrupção no microcontrolador pode fazer de forma interna ao chip ou externa.

e) Tipo de disparo: é a interrupção programada do microcontrolador externo ao chip, pode ser por nível (0 ou 1), borda (subida ou descida) ou por combinação de ambos.

2.7.1.2 Interrupções na família 8051

a) Interrupções externas $\overline{INT0}$ (endereço B2h): É um pino de interrupção que deve ser habilitado via software.

b) Interrupção externa $\overline{INT1}$ (endereço B3h): Semelhante o INT0.

c) Interrupção interna gerada pelo *TIMER/COUNTERS 0*: É uma interrupção interna gerada pelo *TIMER_0*.

d) Interrupção interna gerada pelo *TIMER/COUNTERS 1*: É uma interrupção interna gerada pelo *TIMER_1*.

e) Interrupção pela serial: Interrupção gerada pelo periférico Serial.

O microcontrolador da família 8051 é não vetorada, ou seja, tem endereço de interrupção fixo.

Na tabela 2.8 mostra a descrição de interrupção e o endereço de desvio de interrupção.

Interrupção	Tipo	Endereço de desvio da Interrupção
Reset	Externa – RST	0000h
INT 0	Externa – Pino P3.2	0003h
INT 1	Externa – Pino P3.3	0013h
TIMER/COUNTER_0	Interno – Periférico	000Bh
TIMER/COUNTER_1	Interno – Periférico	001Bh
SERIAL	Interno – Periférico	0023h
**TIMER/COUNTER_2	Interno Periférico	002Bh

** Só no microcontrolador 8052.

Tabela 2.8- Descrição de interrupção e o endereço de desvio de interrupção.
(NICOLSI, 2005, p.146).

Na figura 2.21 mostra o processo de interrupção não vetorada.

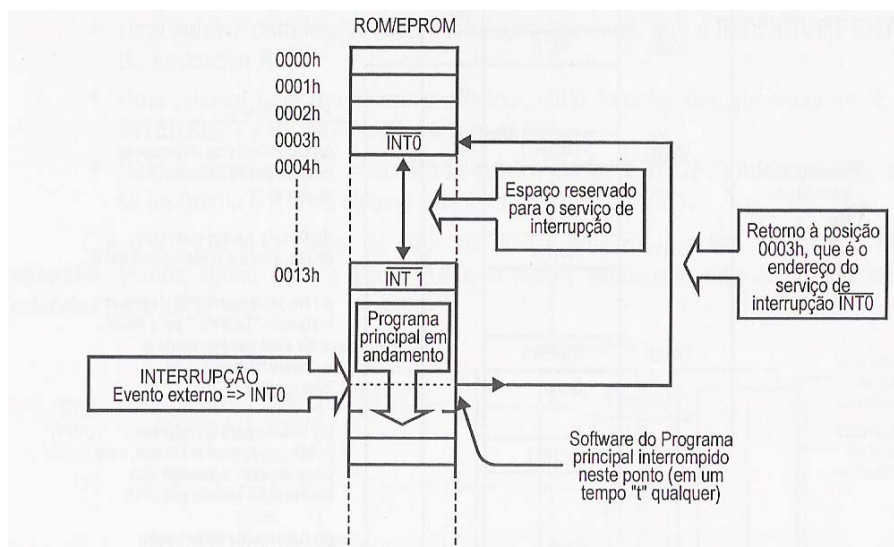


Figura 2.21 – Processo de Interrupção não vetorada. (NICOLSI, 2000, p.147).

A figura 2.22 mostra a interrupção na $\overline{\text{INT0}}$, na ROM/EPROM posições livres de memória, desde endereço 0003h até 0012h, após este endereço inicia-se o endereço $\overline{\text{INT1}}$ (0013h) e mostra o caminho desta interrupção até a inicialização do programa no microcontrolador.

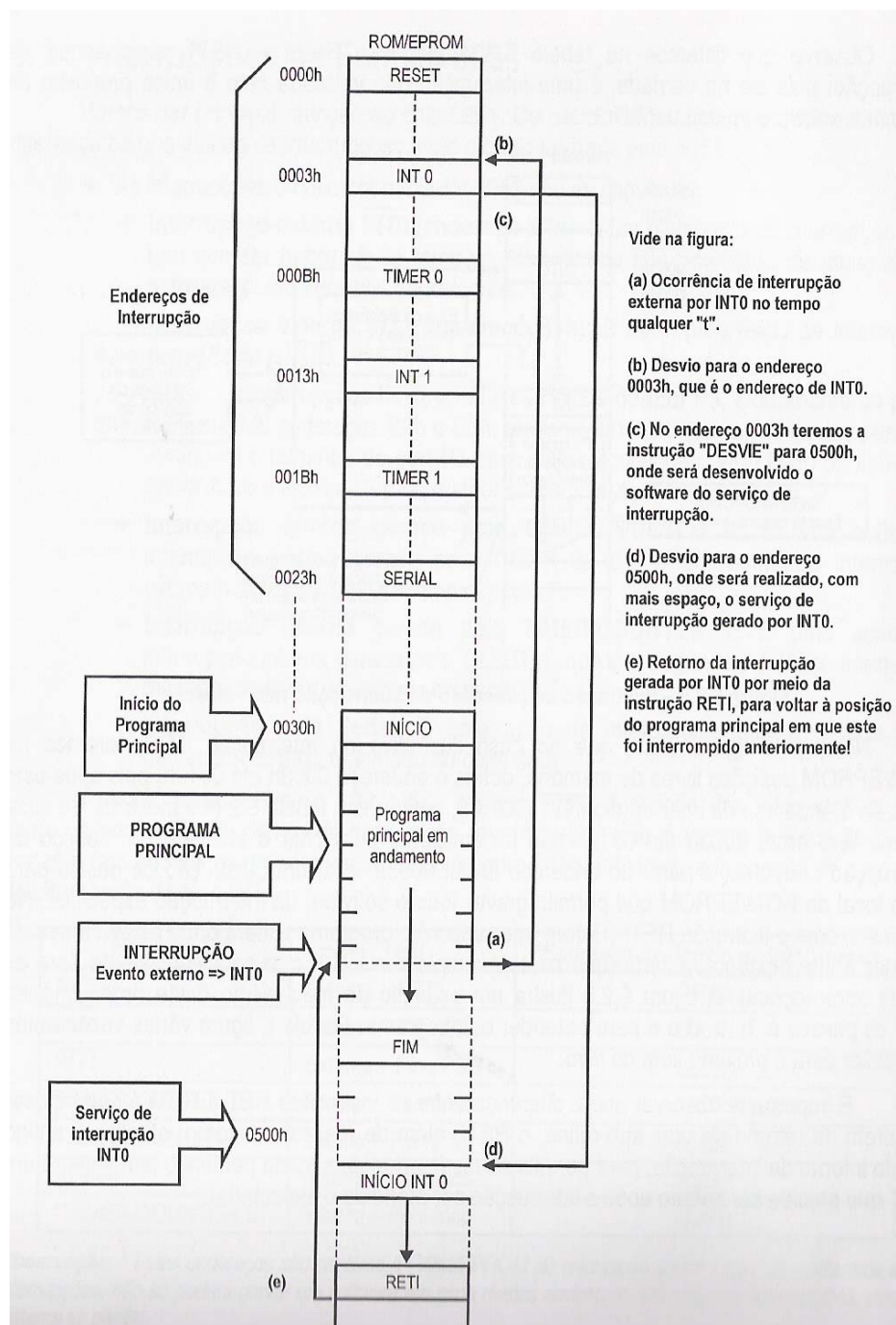


Figura 2.22 – Interrupção na $\overline{\text{INT0}}$. (NICOLSI, 2000, p.148).

A instrução *RETI* é o retorno de uma sub-rotina, mas além deste papel ele atualiza o antigo estado interno para que ocorram novas interrupções.

2.8 Timer

No microprocessador da família 8051 existem dois *timers*. O *timer* é um periférico interno, é conhecido como *TIMER/COUNTERS*.

Timer é um grupo de *flip-flops* em arranjo de “divisor por dois”. O *timer* é acionado pelo mesmo *clock* do microprocessador, só que este *clock* é dividido por 12 antes de entrar no timer.

O *flag* é um sinalizado da contagem do *flip-flop*, ele tem a função de indicar, em um estágio de “*n*” *flip-flop* que a contagem chegou ao fim, tendo assim um “estouro da contagem”.

O *clock* externo conhecido também como *TIMER/COUNTERS* pode ser programados em 8 *bits* ou até 16 *bits*. Estes tipos de *clock* externos são aplicados para gerar base fixa de tempo, medição de tempo, contagem de eventos externos e/ou para gerar a base de tempo do periférico *SERIAL*.

Existem dois registradores de programação especial usados para programar o *timer* externo que são os *TCON* e *TMOD*.

2.8.1 Modos de trabalho do timer

Existem dois *TIMER/COUNTERS*, cada *TIMER* é programado separadamente em quatros modos de trabalho, que são:

1. MODO 0: Contador com capacidade de 13 *bits*.
2. MODO 1: Contador com capacidade de 16 *bits*.
3. MODO 2: Contador com capacidade de 8 *bits* e *auto-reload*.
4. MODO 3: Contador misto, utilizado para aplicações especiais.

2.8.2 Modo 0 (13 bits)

A figura 2.23 mostra o *TIMER/COUNTERS* em modo 0.

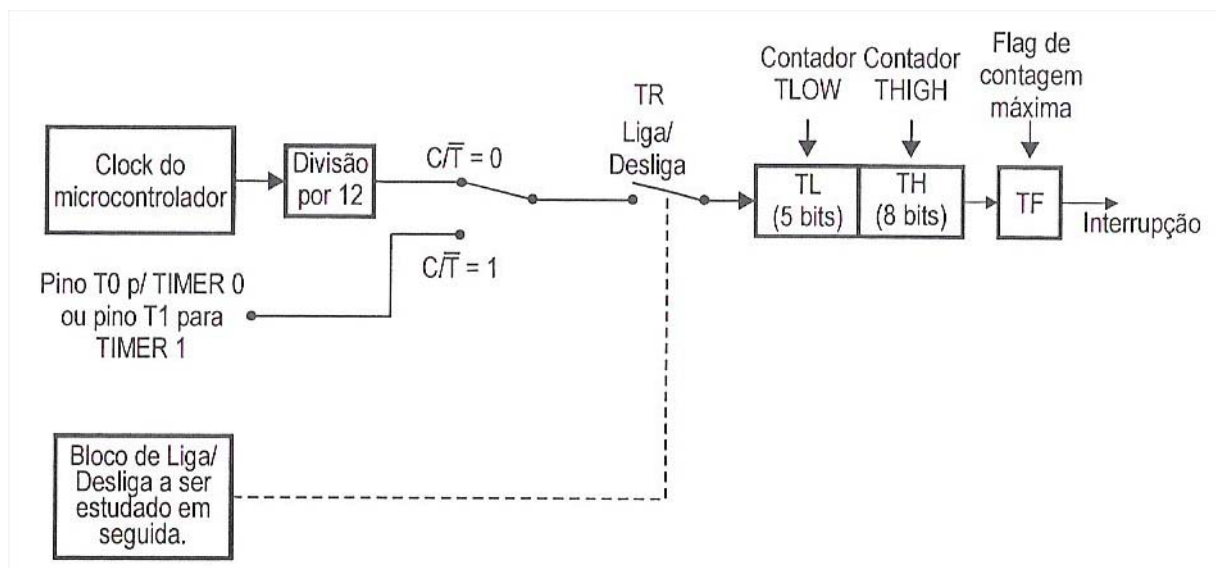


Figura 2.23 – *TIMER/COUNTERS* em modo 0. (NICOLOSI, 2000, p.160).

O TLOW possui 3 bits que não são usados, produzindo assim um total de contagem de até 13 *bits*.

O bit $\overline{C/T}$ no registrador *TMOD*, permite a programação na base de tempo do “clock do microcontrolador dividido por 12”, ou “contagem externa” para o *TIMER/COUNTERS_0*, que é conhecido como T0 (P3.4 do microcontrolador) ou o *TIMER/COUNTERS_1*, que é conhecido como T1 (P3.5 do microcontrolador).

O bit *TR* é conhecido como uma chave de liga/desliga através de software. O flag *TF* é um flag de contagem máxima, que faz com que o contador conta “para cima” e no “estouro de conta”. Para contar 100 pulsos deve carregar os registradores TLOW e THIGH com o número de “contagem máxima – 100”.

2.8.3 Modo 1 (16 *bits*)

A figura 2.24 mostra o *TIMER/COUNTERS* em modo 1.

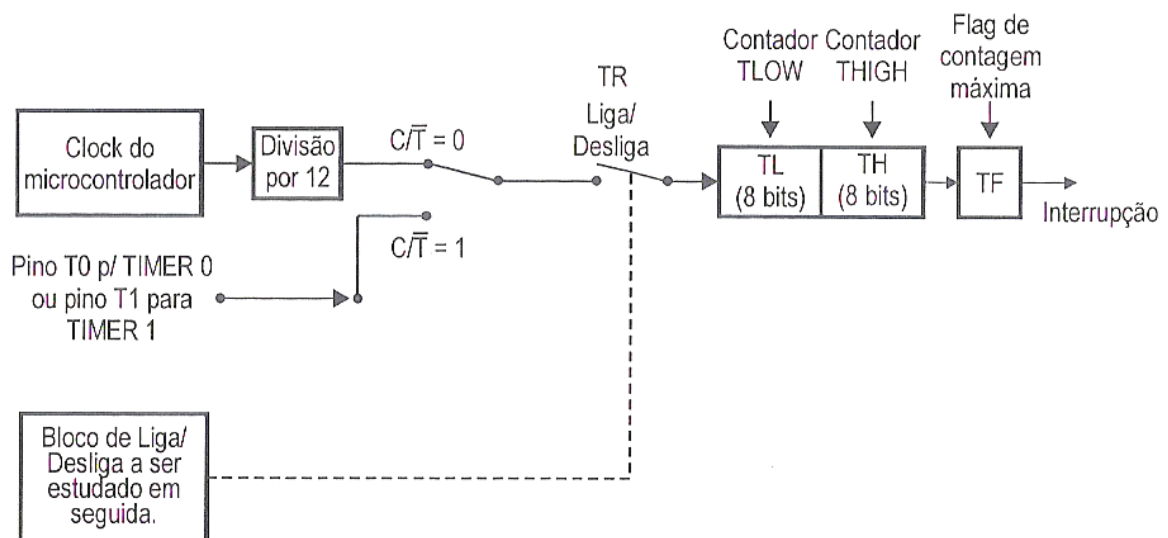


Figura 2.24 – *TIMER/COUNTERS* em modo 1.(NICOLASI, 2000, p.161).

O modo 1 tem a mesma funcionalidade que o modo 0, exceto pelo fato do modo 1 ser 16 *bits*.

2.8.4 Modo 2 (8 *bits* com recarga automática).

A figura 2.25 mostra o *TIMER/COUNTERS* em modo 2.

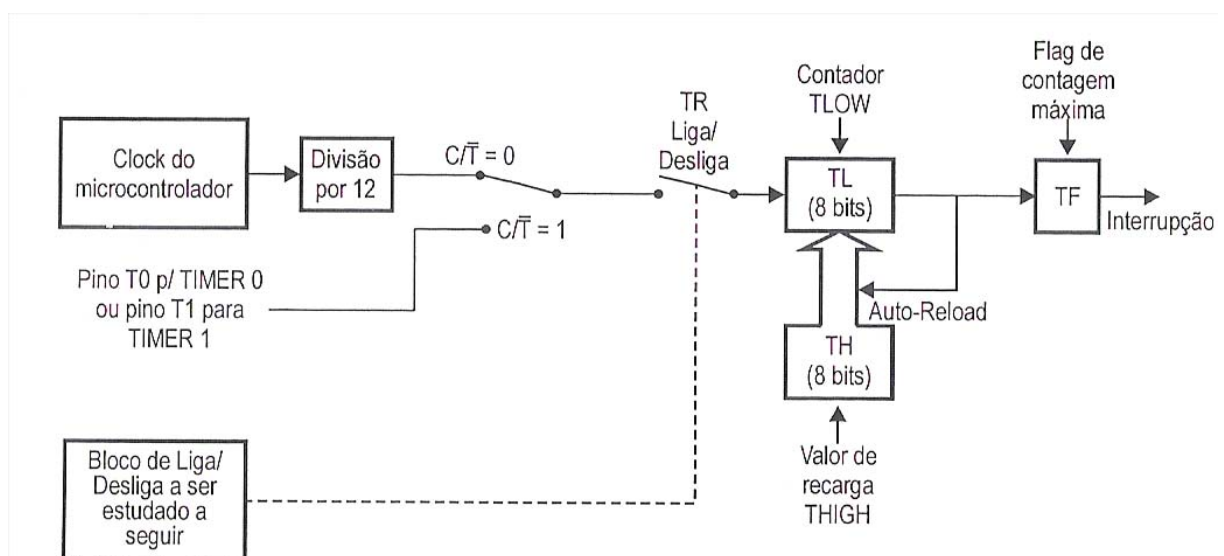


Figura 2.25 – *TIMER/COUNTRS* em modo 2.(NICOLASI, 2000, p.162).

No modo 0 e modo 1 a cada estouro de carga precisava parar o timer e recarregá-lo com o número calculado de contagem. No modo 2 não precisa

desta parada do timer, existe neste modo um processo automático de recarga de número, chamado de *Auto-reload*. Este modo só carrega o registrador TLOW como contador (8 bits) e o THIGH fica registrado como *Auto-Reload*.

2.8.5 Modo 3 (8 bits mistos)

Este modo é conhecido como “modo especial”, o *TIMER/COUNTERS_0* é dividido em dois contadores isolados de 8 bits. Portanto, se neste modo ganhar mais um contador, os flags TF e TR serão emprestados do *TIMER/COUNTERS_1*.

A figura 2.26 mostra o *TIMER/COUNTERS* em modo 3.

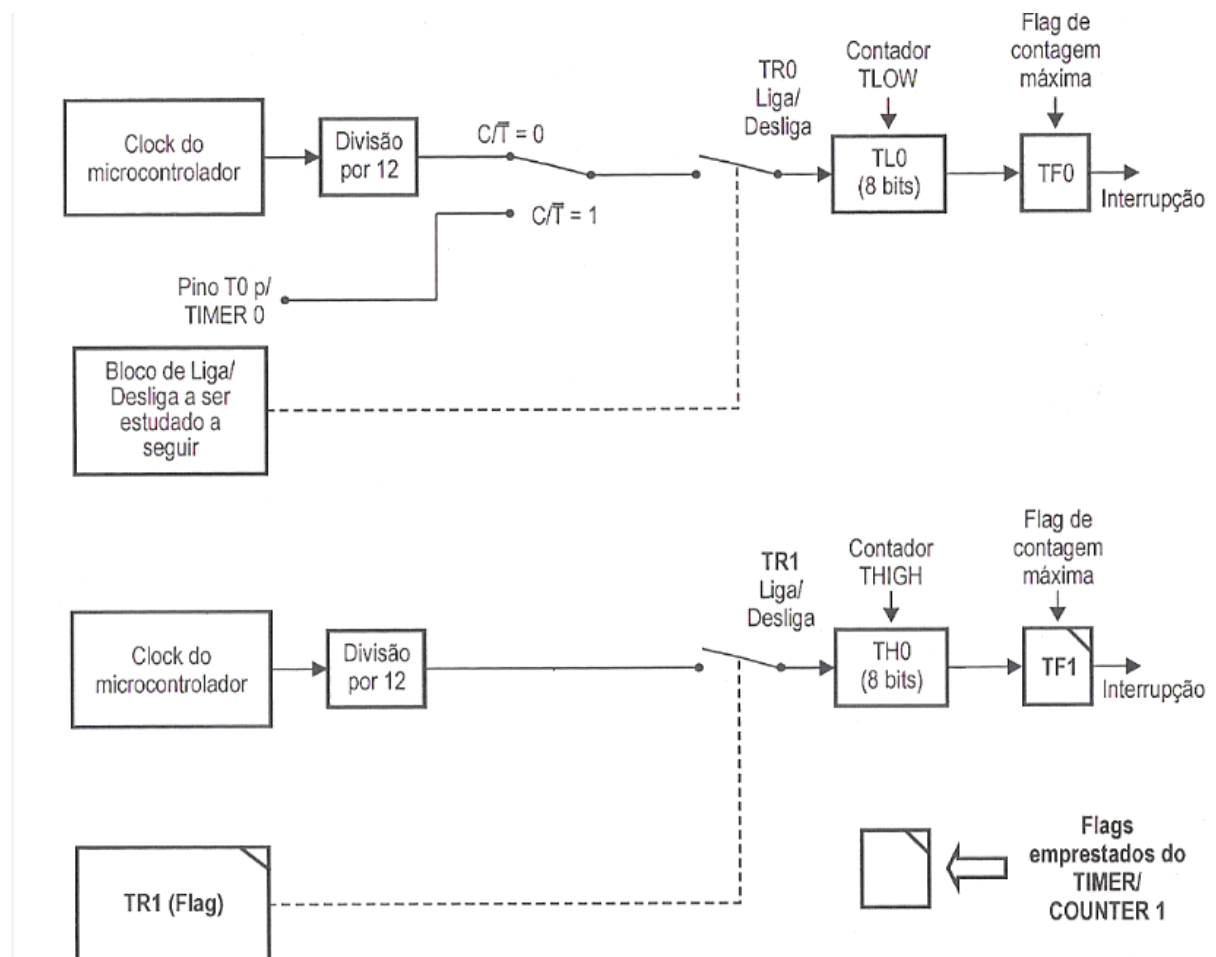


Figura 2.26 – *TIMER/COUNTERS* em modo 3. (NICOLSI, 2000, p.163).

O *TIMER/COUNTERS_1* pode ser usado ainda para gerar o *baud rate* para o periférico da SERIAL.

2.9 Serial

O periférico Serial permite a comunicação bidirecional entre máquina apenas por dois fios, e também, permite a comunicação através de Modem(transferir e receber dados via sistema de telefonia tradicional).

O nome do periférico é Serial devido o modo de transmissão dos dados, estes dados são transmitidos um *byte*, *bit* por *bit*, em seqüência preestabelecida e pré-programada, até que a outra ponta receba um novo *byte*, igual ao original transmitido.

O *baud rate* serve para que as frequências de transmissão e recepção sejam as mesmas.

2.9.1 Modos de comunicação

Os modos de comunicações são de dois tipos:

- a) Serial Assíncrona;
- b) Serial Síncrona;

Serial Síncrona utiliza uma saída como envio ou recepção de dados e necessita de um sincronismo para que o receptor possa verificar que é *bit* “0” e quem é *bit* “1” na comunicação.

A figura 2.27 mostra o processo de transmissão e reconhecimento dos bits em comunicação serial síncrona.

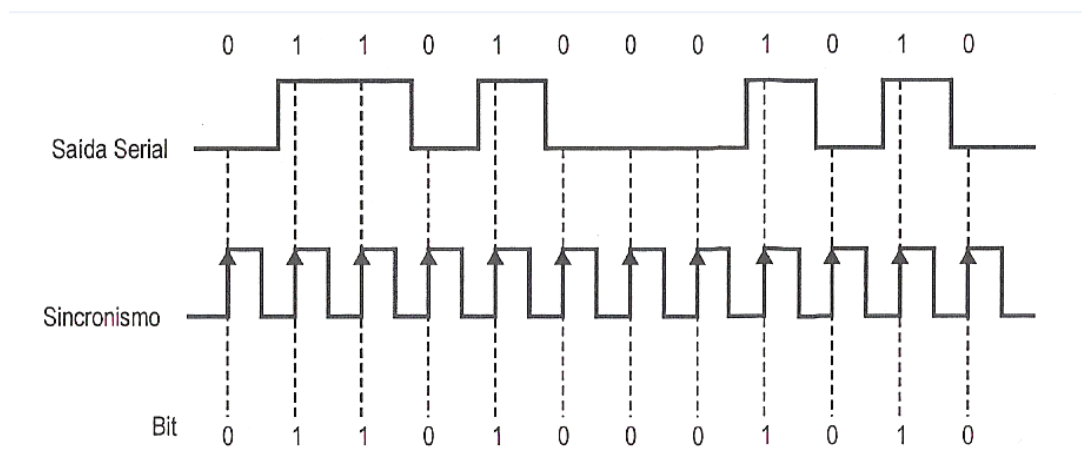


Figura 2.27 – Comunicação da *Serial*.(NICOLSI, 2005, p.59).

A comunicação serial assíncrona não depende do sinal de sincronismo, ou seja, esta comunicação gera um padrão de comunicação de cada *byte* transmitido: cada *byte* possui um *bit* de início (*start*), transição de 1 para 0, e um *bit* de fim (*stop*), transição de 0 para 1, finalizando o processo, então reinicia o processo, aguardando um próximo *bit* de início.

Os microcontroladores da família 8051 possuem um modo de comunicação síncrona (modo 0) e três modos de comunicações assíncronas (modo1, modo 2 e modo 3).

2.9.2 O sistema de transmissão e recepção.

O microcontrolador 8051 possui duas palavras de atuação: de controle (*SCON*) e de transmissão e recepção (*SBUF*). O registrador *SBUF* possui dois registradores: um de transmissão e outro de recepção.

A figura 2.28 mostra a atuação do *SBUF* na *Serial*.

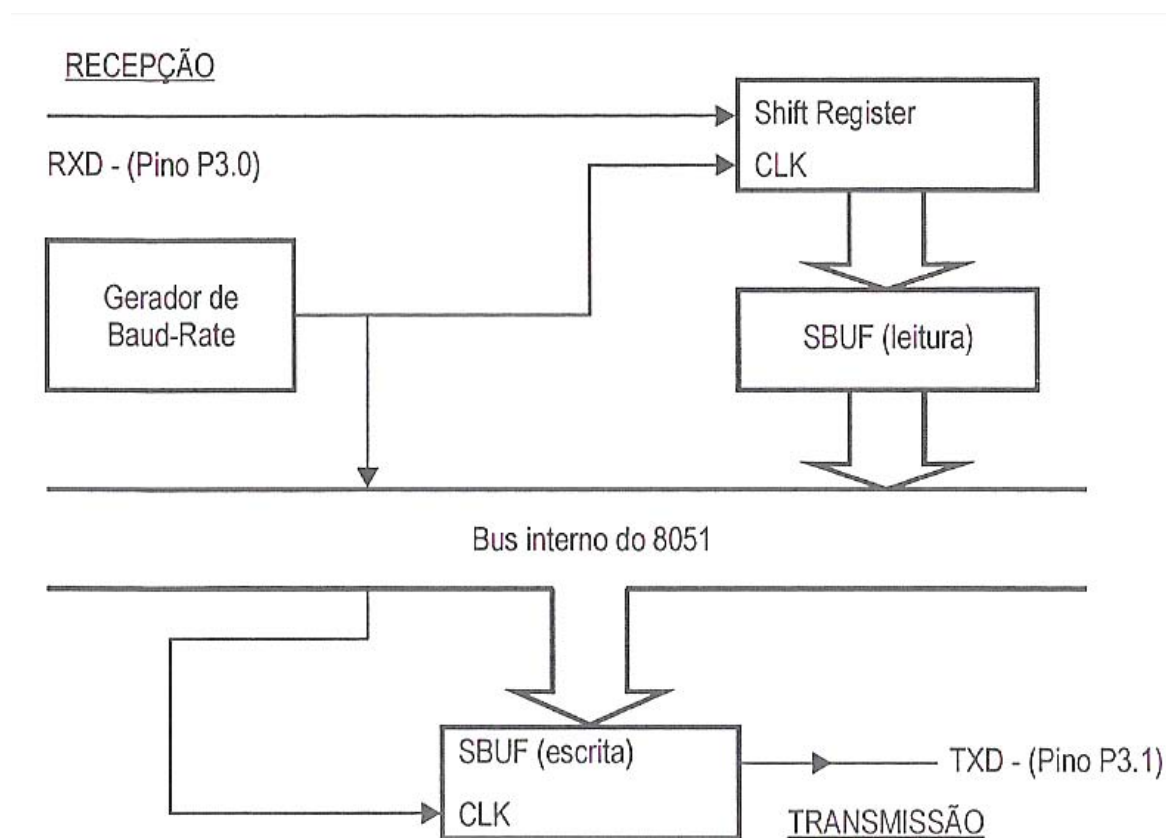


Figura 2.28 – *SBUF* na *Serial*.(NICOLSI, 2005, p.58).

2.9.3 A palavra de controle *SCON*

SCON é o registrador que permite programar e controlar todos os modos do periférico da serial juntamente com o SBUF.

A figura 2.29 mostra o que *SCON* consiste.

NOMES:									END
SCON	SM0	SM1	SM2	REN	TB8	RB8	TI	RI	98h
END DE BIT	9F	9E	9D	9C	9B	9A	99	98	(BYTE)

Figura 2.29 – Controle da *SCON*. (NICOLOSI, 2000, p.178).

Os *bits* SM0 e SM1, combinados, mostram os quatro modos possíveis de operações. O *bit* SM2 é utilizado em multiprocessadores. O bit *REN* permite iniciar a recepção de dados. Os bits TB8 e RB8 permitem enviar um nono *bit* junto com cada *byte* transmitido. Os *bits* TI e RI são *flags* de interrupção para transmissão, o TI é para transmissão e o RI para recepção.

2.10 Temporizador

O temporizador é um dispositivo capaz de medir o tempo, sendo um tipo de relógio especializado. Ele pode ser usado para controlar a sequência de um evento ou processo. Temporizadores podem ser mecânicos, digitais, eletromecânico, ou mesmo programas de computadores, caso o computador tenha relógio.

Na arquitetura dos microcontroladores, os temporizadores podem ser utilizados para os mais diversos fins, por exemplo, para gerar sinais de clock para outros periféricos do chip, calcular intervalos de tempo ou medir período de sinais. No meio acadêmico os temporizadores mais utilizados na arquitetura dos microcontroladores são MCS51 e o PIC 16, estão presentes temporizadores de 8 *bits* e 16 *bits*, podendo ser programado pelo usuário de distintas formas.

3. Construção do Protótipo

Este capítulo mostra o desenvolvimento do projeto, que foi dividido em duas partes: a primeira parte é o circuito lógico, onde foram apresentados os componentes que o compõe e o funcionamento de cada um. Na segunda parte são mostrados o funcionamento do circuito e suas características em uma residência.

A figura 3.1 mostra o protótipo implementado neste projeto.

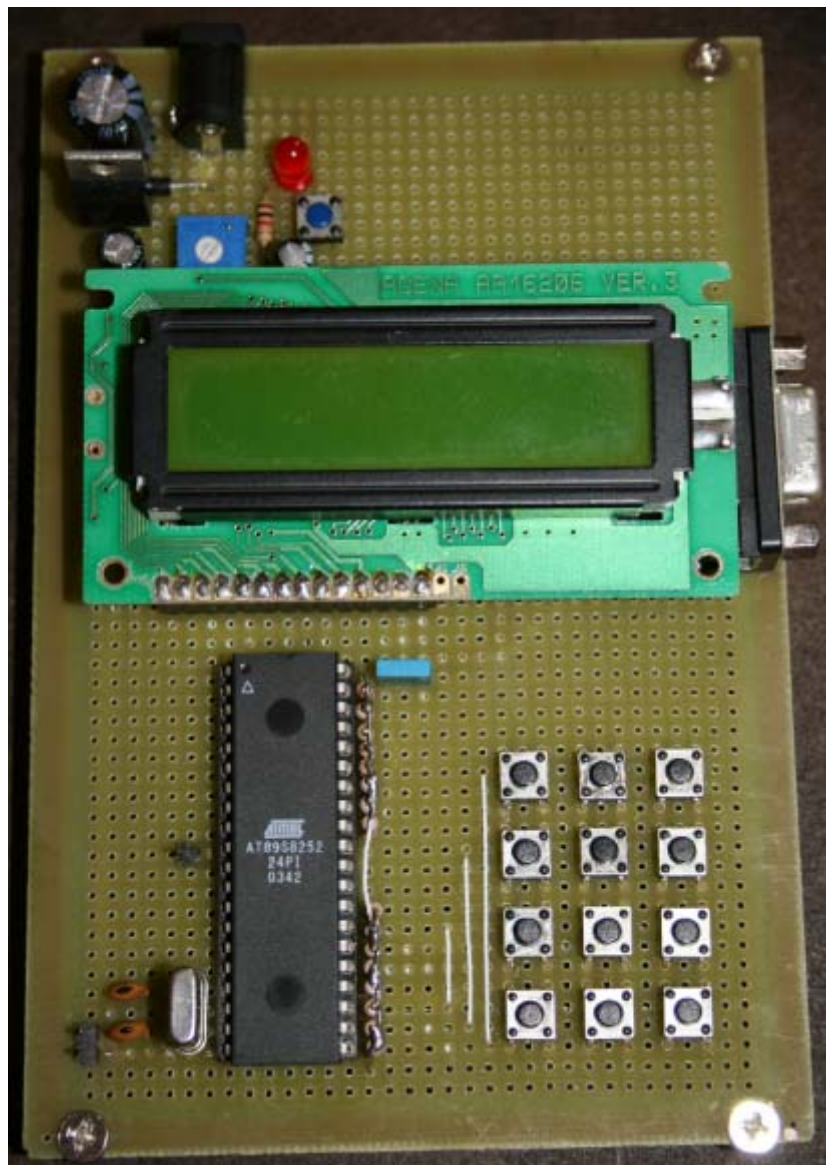


Figura 3.1 – Protótipo implementado.

Os componentes utilizados neste projeto foram escolhidos visando qualidade, rapidez, baixo custo e simplicidade. Na tabela 3.1 é mostrado o custo de cada componente e ferramenta utilizada para o desenvolvimento do projeto.

Custos dos Componentes e Ferramentas		
Componentes e Ferramentas	Quantidade	Preço
Conector DC	1	R\$ 3,00
7805	1	R\$ 1,20
Capacitor Eletrolítico 470 micro F/25 Volts	1	R\$ 0,50
Capacitor Eletrolítico 100 micro F/25 Volts	1	R\$ 0,30
Capacitor Poliéster 100Nf/25 Volts	3	R\$ 1,50
Resistores 10 Kohms/1/8(tamanho)	18	R\$ 1,80
Cristal 11,359 MHz	1	R\$ 1,60
Capacitor de cerâmica 27 PF	2	R\$ 0,60
Trim-pot 5 K	1	R\$ 3,00
Microcontrolador	1	R\$ 60,00
Display	1	R\$ 40,00
Botão do teclado	12	R\$ 2,40
Total	44	R\$ 115,90

Tabela 3.1 – Custo do protótipo.

3.1. Hardware

3.1.1. Microcontrolador

O microcontrolador utilizado é da família 8051, AT89S8252. Neste protótipo suas principais funções são utilizadas, pois este microcontrolador tem a capacidade de trabalhar com palavras de 8 bits, de alta performance e baixo custo.

As principais configurações deste microcontrolador são (NICOLASI, 2005, p.24):

- 64 *Kbytes* de memória para *DATA (RAM)* e *CODE (ROM)*.
- 256x8 *bytes* de memória *RAM*, dividida em área de uso geral e registradores especiais.
- Dois *timers/counters* de 16 *bits*.
- Uma porta serial programável (*UART*).
- Interface para memória externa com capacidade de 64 *Kbytes* de endereçamento externo para *ROM* e 64 *Kbytes* de endereçamento externo para *RAM*.
- Quatro portas de *I/O*.
- Seis possibilidades de interrupções com dois grupos de prioridades.

Neste protótipo as principais características são utilizadas, sendo que somente a memória externa não é utilizada neste projeto.

O microcontrolador possui uma memória RAM e ROM de 64 Kbytes para armazenamento de dados.

A memória RAM tem a função de armazenar dados. Os dados armazenados na memória são os caracteres do teclado, a forma que as palavras no *display* LCD são escritas e o acionamento da lâmpada.

A memória ROM tem a função somente de leitura. Cada tecla a ser digitado do teclado será lida através da memória. Leitura de cada posição e cada caractere que será apresentado no *display* LCD.

As portas utilizadas de *I/O* são as portas P0, P1, P2 e P3. A utilização de cada porta no protótipo será descrita de acordo com cada componente.

3.1.2. Porta Serial

A comunicação entre o computador e o microcontrolador é realizada através de uma porta serial. O software é responsável por enviar os dados do *display* e do teclado para o microcontrolador e ler a resposta do mesmo para interface de comunicação do notebook por esta porta.

A porta Serial utilizado neste protótipo será RS-232, modelo DB 9, por ser a interface mais utilizada para comunicação com os microcomputadores.

A porta Serial tem a função de transmitir os dados do computador para o microcontrolador. Para que a transmissão seja realizada é necessária os circuitos integrados MAX 232 e o SN74HC126N.

3.1.2.1. Comunicação com RS-232

O RS (*Recommended Standart*) foi criado nos anos 60 como uma padronização de interface serial para facilitar a interconexão dos terminais e dos equipamentos de comunicação de dados. O padrão RS-232 especifica as tensões, temporizações e funções dos sinais, um protocolo para troca de informações, e as conexões mecânicas.

No RS-232 os pinos mais utilizados são três, sendo um com função de envia, um com a função de receber dados e outro com a função de circuito comum conforme a tabela 3.2. Outros pinos nesta interface são imprevisíveis. (TAFENER,1996 *apud* CALHEIROS,2007).

Pinos	Função
Pino 2	Pino para transmissão
Pino 3	Pino para recepção
Pino 5	Circuito Comum

Tabela 3.2 – Pinos de comunicação Serial

Os pinos da porta Serial utilizado neste protótipo são: Pino 2, Pino 3 e Pino 5. O pino 2 tem a função de transmitir os dados do software utilizado no

notebook para o microcontrolador. O pino 3 tem a função de receber os dados do software utilizado no notebook para o microcontrolador. O pino 5 tem a função de terra do RS-232.

A respeito das características dos sinais elétricos, o RS-232 define atualmente 4 níveis lógicos.

Sinais com tensão entre -3 volts e -25 volts com relação ao terra (pino 5) são considerados níveis lógico “1” (condição marcada), e tensões entre +3 volts e +25 volts são considerados níveis lógico “0” (condição espaço). A faixa da tensão entre -3 volts e +3 volts é considerado uma região de transição para o qual o estado do sinal está indefinido. (CANZIAN, http://www.coinfo.cefetpb.edu.br/professor/leonidas/irc/apostilas/comun_serial.pdf, ACESSO EM 08/06/08).

3.1.2.2. Conversor USB-Serial

O conversor USB-Serial tem como objetivo conectar acessórios e periféricos na porta USB do notebook, tornando os periféricos Plug&Play permitindo a velocidade de transferência de 1 Mbyte.

Este conversor é utilizado para fazer a comunicação do notebook, pois o notebook em uso não possui porta serial, com o circuito integrado (CI) MAX 232 e o CI SN74HC126N.

Para que a linguagem de máquina (binária) do computador seja transmitida para a linguagem de programação do microcontrolador (hexadecimal) é necessária a conexão do conversor. O conversor está na figura 3.2.



Figura 3.2 – Cabo conversor USB-Serial

3.1.3. Circuito Integrado (CI) MAX 232

O circuito integrado MAX 232 é um conversor de nível TTL (Lógica Transistor-Transistor) para RS 232. Para que equipamento conectado em RS-232 é necessário transformar nível TTL (0 à 5 volts) em RS-232 ou transformar de RS-232 em nível TTL.

A transformação do nível TTL para RS-232 utiliza o *driver* 1488 e a transformação do nível RS-232 para TTL utiliza o *receivers* 1489.

Este CI possui 4 inversores do mesmo tipo, seja eles *drivers* ou *receivers*. O inversor do tipo *driver* necessita de 2 fontes de alimentação +7,5 volts a +15 volts e -7,5 volts a -15 volts. O problema é que somente uma fonte de 5 volts é utilizada.

O circuito MAX 232 possui uma ótima rejeição de ruídos e é mais robusto a descargas e curtos circuitos.

O MAX 232 inclui um circuito “*charge jump*” capaz de gerar uma tensão de -10 volts a +10 volts a partir de uma fonte de +5 volts, bastando apenas alguns capacitores externos de 10 microFaraday e 50 volts. O CI possui ainda 2

inversores do tipo *drivers* e 2 inversores do tipo *receivers* no mesmo encapsulamento.

O CI MAX 232 está conectado diretamente ao CI SN 74HC126N.

A figura 3.3 mostra o CI MAX 232.



Figura 3.3 – MAX 232

3.1.4. Circuito Integrado (CI) SN74HC126N

O SN74HC126N tem uma faixa de tensão de 2 volts a 6 volts. Este CI possui 3 estados de saída que são: “0”, “1” e alta impedância, no qual pode variar de “0” a “1”. Esta variação de tensão só é dada se estiver nos 3 estados de saída.

Este tipo de CI tem a função de amplificar a potência.

Neste protótipo a função deste CI é para a comunicação do RS-232 com o microcontrolador. Após o programa ser transmitido para o microcontrolador, este CI tem a função de habilitar o *reset* automaticamente.

O SN74NHC126N está conectado ao microcontrolador através da porta P1.

A figura 3.4 mostra o CI SN74HC126N.



Figura 3.4 – SN74HC126N

3.1.5. Display LCD de 2 linhas e 16 colunas

O *display* é um periférico de saída que serve para a leitura de gráficos ou caracteres.

O *display* utilizado neste protótipo será de 2 linhas e 16 colunas, devido a utilização de caracteres para informações dos dados.

O *display* neste protótipo tem como função principal informar ao usuário para que seja digitada a senha *default*, esta senha ao ser informada corretamente, informará para que o usuário digitar o tempo em que aquele ambiente residencial será desligado.

As figuras 3.5 e 3.6 mostram respectivamente a foto e o fluxograma do *display*. O acionamento do *display* pelo microcontrolador é realizado pela porta P0 (vide capítulo 2, página 31).

A figura 3.5 mostra o *display* utilizado no protótipo.

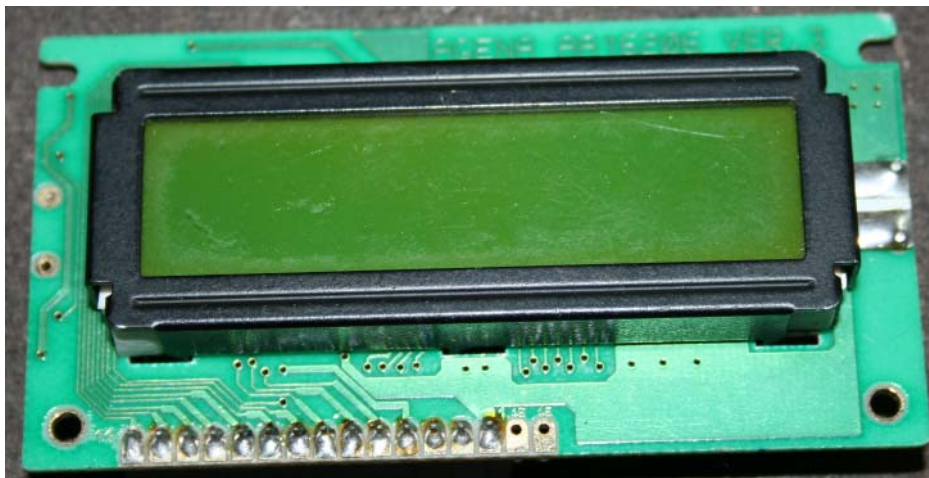


Figura 3.5 – Display LCD 2X16

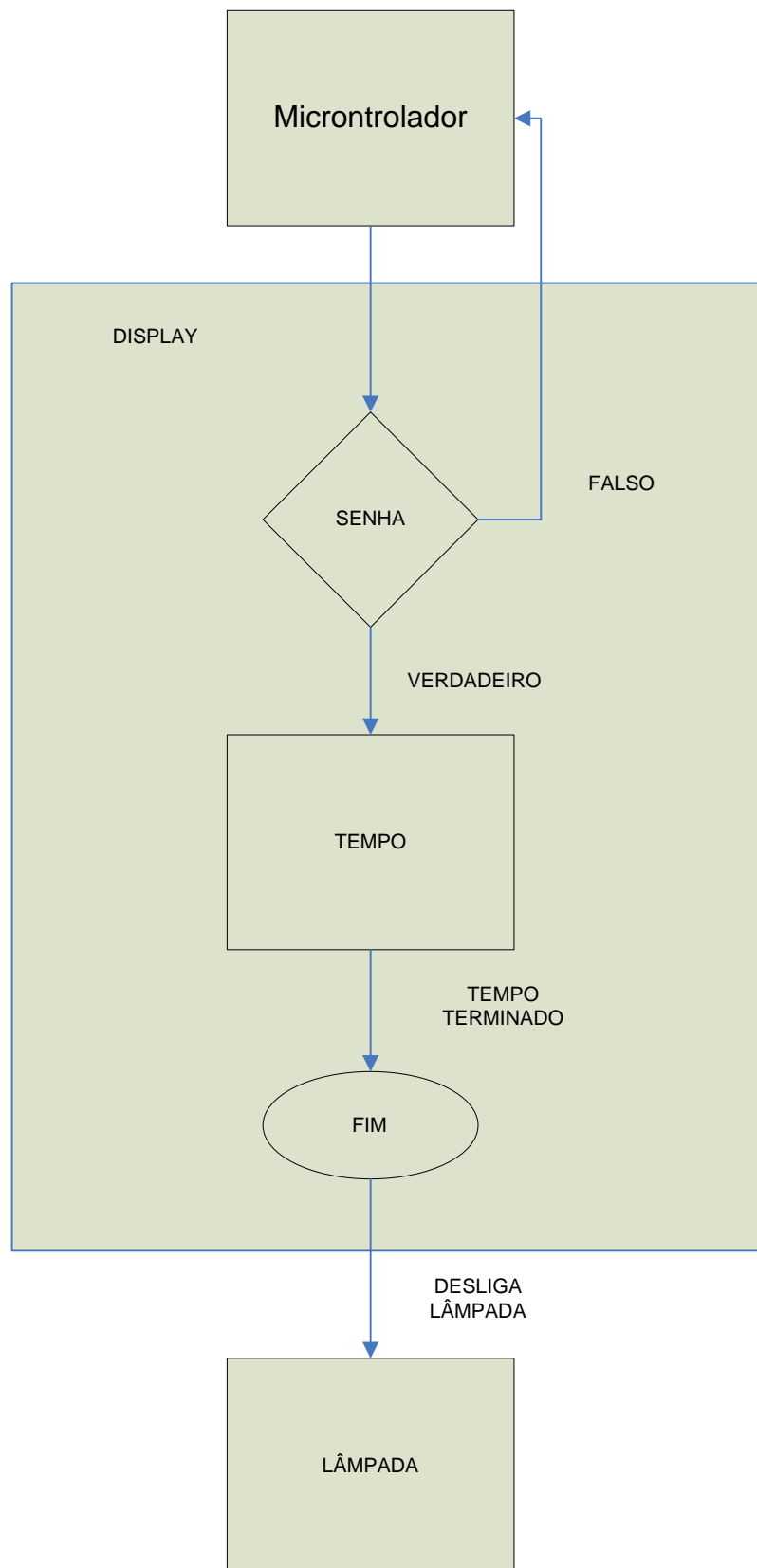


Figura 3.6 – Fluxograma do *Display*

O código fonte do *display* é mencionado do apêndice. O código fonte de inicialização do *display* está mencionado no apêndice página 92. O código fonte da sub-rotina do *display* está mencionado no apêndice na página 93. O código fonte da rotina de escrever no LCD está mencionado no apêndice na pagina 96. O código fonte da rotina para escrever na segunda linha do *display* está mencionado no apêndice na pagina 98.

3.1.6. Teclado numérico matricial 4x3.

O teclado é um periférico de entrada que serve para escrita de gráficos ou caracteres.

O teclado utilizado neste protótipo foi um teclado matricial de 4 linhas e 3 colunas. Este foi baseado em um teclado de telefone convencional. Este teclado tem algumas teclas com funções diferentes do teclado telefônico convencional como é o caso da tecla “*” que tem a função de limpar a tela, reinicializar o contador do relógio e de sair da tela de inicialização mostrado no *display* e a tecla “#” tem a função de entrada, ou seja, para que inicie o tempo determinado pelo usuário é necessário que o mesmo pressione a tecla. Ao pressionar qualquer tecla do teclado o microcontrolador realiza um *debaunce* na porta correspondente aquela tecla pressionada. O *debaunce* é utilizado para que haja uma estabilidade do nível lógico após o fechamento do contato elétrico das teclas, ou seja, evita o acionamento várias vezes da tecla indevidamente.

O teclado matricial está conectado diretamente no microcontrolador através da porta P2.

A função do teclado neste protótipo é que o usuário digite sua senha e após a senha, se ela for validada, será digitado o tempo em que cada equipamento elétrico ficará ligado no ambiente residencial.

As figuras 3.7 e 3.8 mostram respectivamente a foto e o fluxograma do teclado matricial.

A figura 3.7 mostra o teclado utilizado no protótipo.

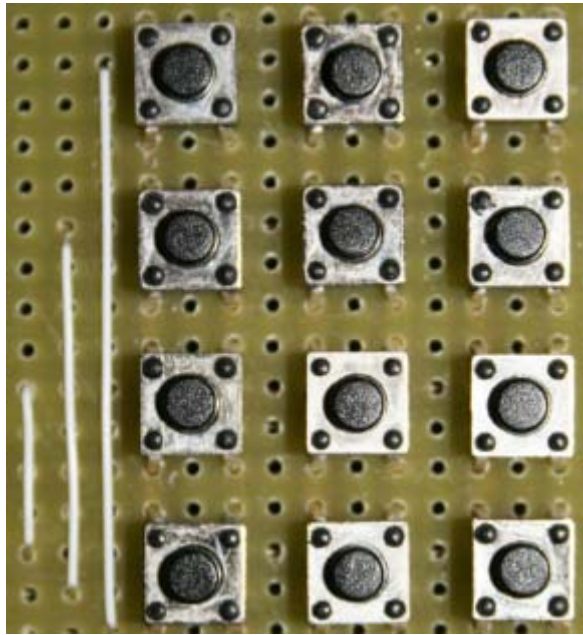


Figura 3.7 – Teclado Numérico matricial 4x3

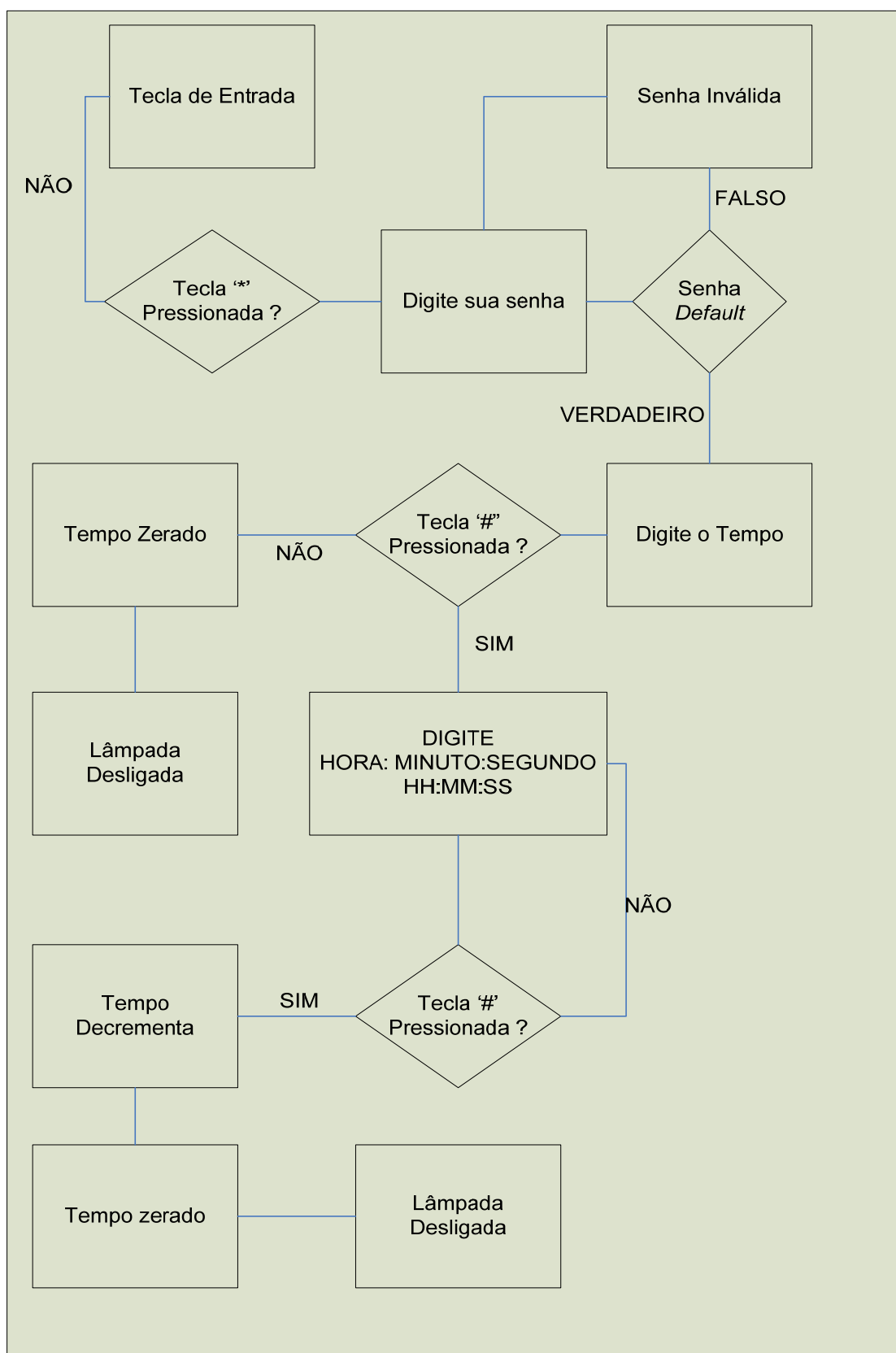


Figura 3.8 – Fluxograma do Teclado

O código fonte do teclado matricial é mencionado do apêndice. O código fonte da leitura do teclado está mencionado no apêndice página 99. O código fonte da rotina do teclado matricial 4x3 está mencionado no apêndice na página 99. O código fonte da rotina de *debounce* está mencionado no apêndice na pagina 101.

3.1.7. PC 817.

O PC817 é um foto acoplador.

O foto acoplador é um CI formado por um led e um fototransístor internamente. A função do foto acoplador é transferir uma informação elétrica entre dois circuitos através de luz, ou seja, sem contato elétrico entre eles.

As funções deste CI no protótipo é aplicar uma tensão nos pinos do led que ao acender, emite uma luz que polariza a base do fototransístor interno, que por sua vez conduz e faz a corrente circular por outro circuito, que no caso é o de acionamento da lâmpada. E outra função é isolar a carga elétrica da lâmpada de 220 volts com a carga elétrica do MC de 5 volts, ou seja, caso haja uma sobrecarga no circuito isolado da lâmpada o foto acoplador impedirá que esta sobrecarga afete o MC.

A figura 3.9 mostra o foto acoplador PC817.

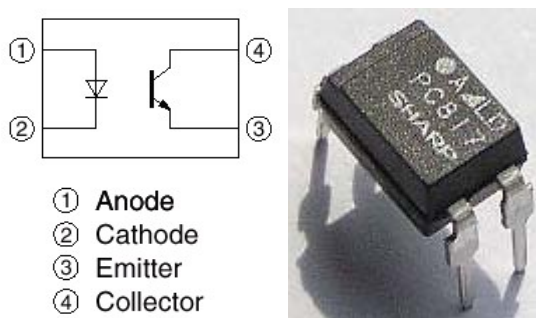


Figura 3.9 – Foto acoplador PC817

3.1.8. Relé

O relé é um componente eletromecânico que tem como função controlar circuito externo de alta corrente a partir de pequenas correntes ou tensões.

O acionamento do relé é realizado por uma corrente que ao circular em uma bobina cria-se um campo magnético que atrai uma serie de contatos fechando ou abrindo circuitos. Ao cessar a corrente da bobina o campo magnético também cessará, fazendo com que os contatos voltem a posição original.

O relé utilizado no protótipo serve para o acionamento e o desligamento do dispositivo elétrico utilizado, que é a lâmpada.

A figura 3.10 mostra a foto de um relé



Figura 3.10 – Foto do relé

3.1.9. Notebook

A utilização do notebook no projeto é para que fosse realizada a instalação dos softwares: Terminal V.25, responsável pela programação do sistema que foi inserido no microcontrolador, o Microflash Electro responsável pela transformação da linguagem de máquina (binária) para a linguagem do

microcontrolador (hexadecimal), o cabo USB-Serial responsável pela transmissão dos dados armazenados no computador (binário) para o microcontrolador (hexadecimal) e o Proteus 7.0 software utilizado para construir o protótipo via software para que depois fosse implementado via hardware.

Para que este tipo de transmissão seja realizado com sucesso é necessário um hardware mais robusto.

A figura 3.11 mostra o notebook



Figura 3.11 - Notebook

Na tabela 12 mostra as especificações do hardware, o notebook, para o funcionamento do protótipo.

Modelo	HP 530
Processador	Intel Core Duo T2400 1.83 Ghz.
Memória Principal	1 Gbytes de RAM DDR 333 Mhz
Memória Secundária	HD de 120 Gbytes
Comunicação externa	2 portas USB 2.0
Sistema operacional	Windows Vista

Tabela 3.3 – Especificações do notebook.

3.2. Software

3.2.1. Sistema operacional

O sistema operacional utilizado é o Windows Vista, versão final, sistema operacional pesado para o processamento do hardware.

Neste sistema foram instalados os seguintes softwares para que fosse feito o protótipo utilizado:

- PROTEUS, versão 7.0;
- TERMINAL, versão 2.5;
- ECLIPSE C/C++;
- MICROFLASH.

3.2.2. Proteus.

O Proteus é um software utilizado para construção do hardware do protótipo.

Neste software foi possível verificar todos os componentes eletrônicos utilizado do protótipo, e fazer testes em cada componente antes da construção definitiva do mesmo.

Este software também foi utilizado para testar a parte de programação do microcontrolador, na linguagem hexadecimal.

A figura 3.12 mostra o software Proteus, esquema eletrônico do protótipo.

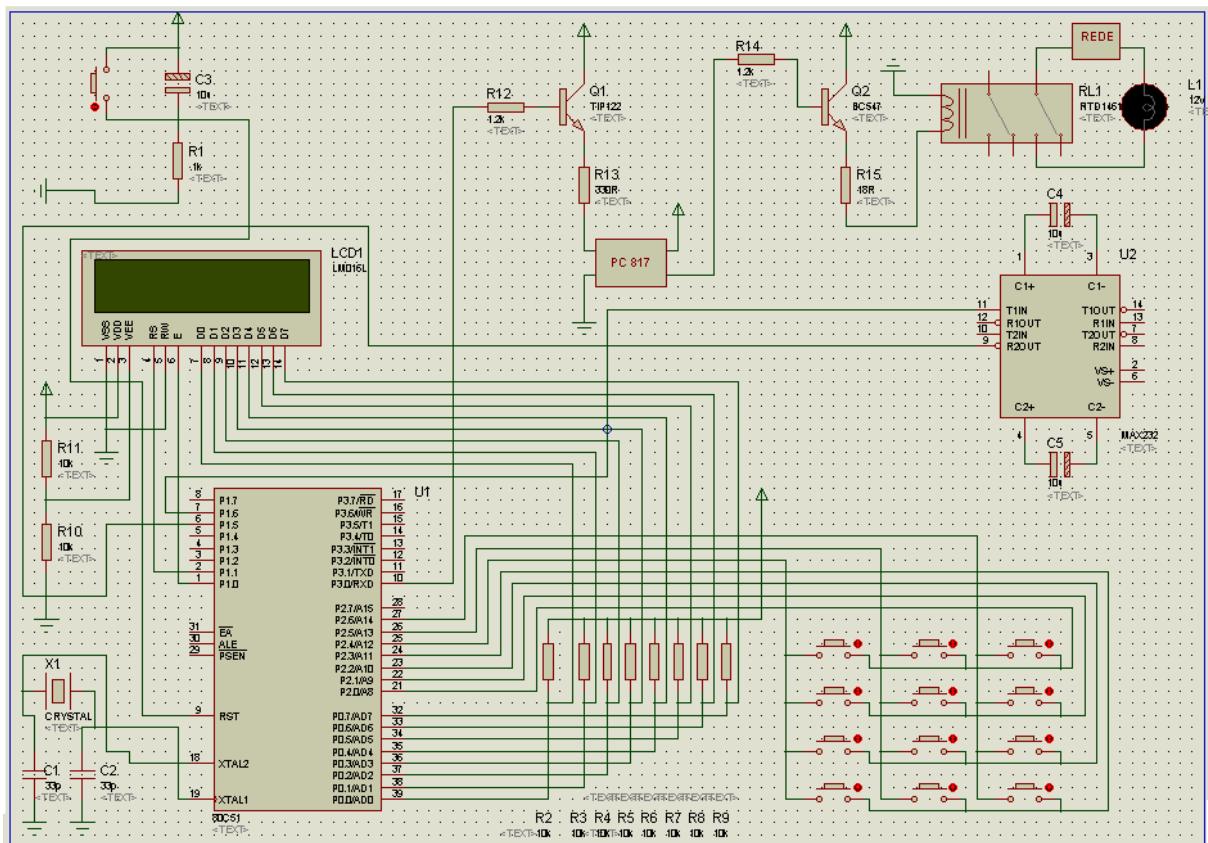


Figura 3.12 – Proteus 7.0

3.2.3. Terminal

O terminal é utilizado para a digitação do código fonte do microcontrolador, na linguagem *ASSEMBLY*.

A linguagem utilizada para a programação do microcontrolador é o *Assembly* por ser uma linguagem de baixo nível - por ocupar menor espaço de memória do microcontrolador.

A figura 3.13 mostra o software Terminal.

Então para que a programação do microcontrolador fosse realizada com sucesso, este software foi abandonado e então a programação foi realizada em *Assembly*, pelo software Terminal.

Figura 3.14 mostra o software Eclipse C/C++.

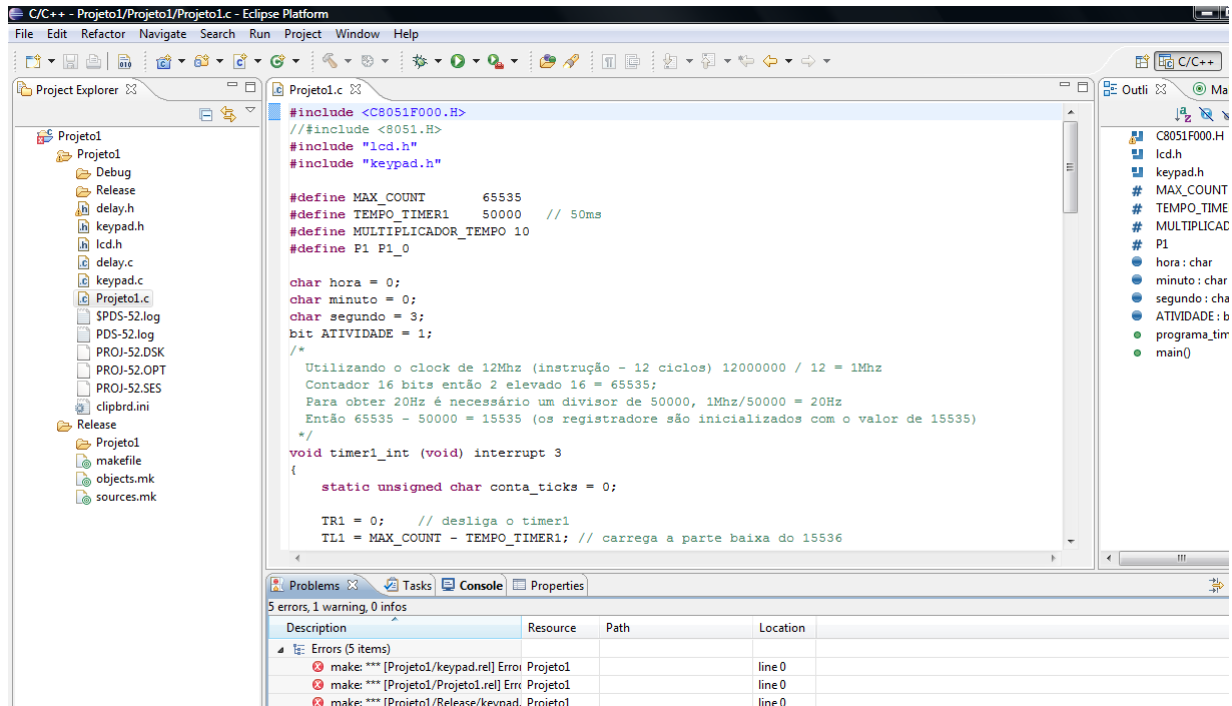


Figura 3.14 – Eclipse C/C++

3.2.5. MicroFlash

O *microflash* é o software utilizado para transformar a linguagem do *Assembler* (.ASM) para a linguagem do microcontrolador (.hex) e para a transmissão dos dados .hex para o microcontrolador.

A figura 3.15 mostra o software *Microflash*.

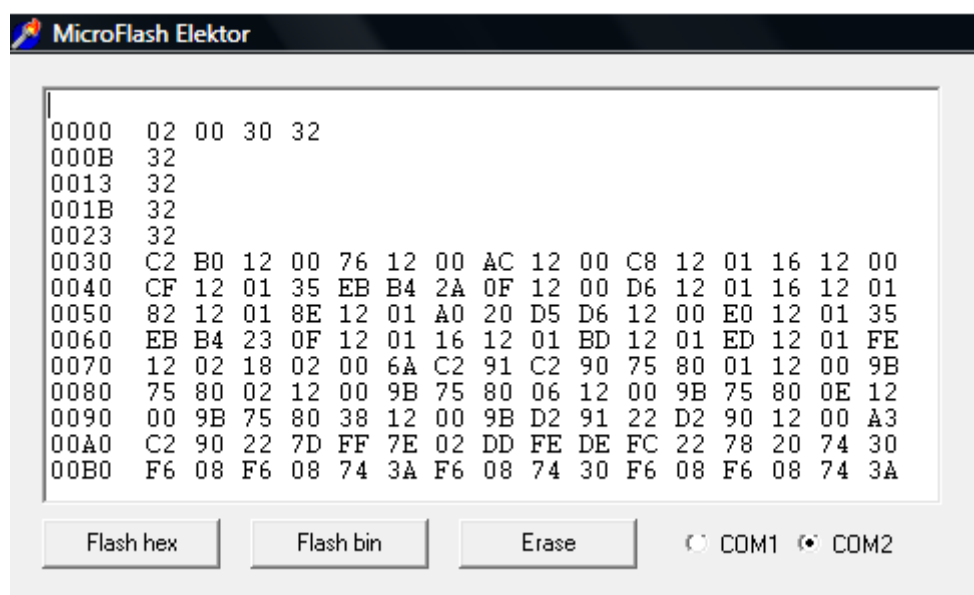


Figura 3.15 – *Microflash.*

4. Conclusão.

A implementação de um protótipo de circuito residencial com temporizador individual por usuário, utilizando um microcontrolador AT89S8252, que tem a função de armazenamento dos dados e comunicações com outros periféricos utilizados, além do *display* LCD, que tem a função de passar as informações visuais dos processos que estão sendo executado pelo usuário, e o teclado, mostrou muito eficiente para os resultados obtidos neste projeto.

O desenvolvimento deste projeto mostrou todo o conhecimento adquirido em circuitos eletrônicos, linguagem de programação e microcontrolador.

4.1. Resultado obtido

O resultado obtido é provar que o circuito pode ser utilizado para uma economia de energia e controlar o tempo que cada pessoa poderá ter em seu ambiente residencial.

4.2. Dificuldades encontradas

Neste trabalho várias dificuldades foram encontradas, nesta subseção todas as dificuldades são relatadas com intuito de contribuir para trabalhos futuros.

As primeiras dificuldades encontradas neste trabalho são de encontrar os componentes eletrônicos, como é o caso do microcontrolador e do *display* LCD em uso neste projeto.

Outra dificuldade encontrada é a comunicação de cada equipamento eletrônico encontrado neste protótipo, a soldagem de cada equipamento e a posição que cada um deve ser colocado na placa para que nenhum problema ocorra como curto circuito ou fazer com que gere uma carga elétrica maior do que cada componente possa suportar para que não queime o componente específico ou outro qualquer que se encontra conectado na placa.

A comunicação entre o notebook e a porta serial do protótipo gera uma certa dificuldade para que sejam passadas as informações da linguagem

binária para hexadecimal, fazendo com que o microcontrolador não receba todas as informações necessárias, gerando assim o travamento do microcontrolador e as vezes, a perda da comunicação da porta serial do protótipo com a porta USB do computador.

A principal dificuldade encontrada neste trabalho é a parte de programação do microcontrolador. A primeira programação foi realizada no software do Eclipse C/C++. Para que este software fosse implementado no computador foi preciso ser instalado um VMware, ou seja, uma máquina virtual no computador, com sistema operacional Windows XP, Serve Pack 2, para que todas as bibliotecas do software fossem reconhecidas, já que o sistema operacional instalado do computador é o Windows Vista. Após a instalação desta máquina virtual no computador, o software Eclipse C/C++ foi instalado para que fosse feito o programa necessário para o microcontrolador.

O programa do microcontrolador, implementado no Eclipse C/C++, ao ser enviado do computador para o microcontrolador, através do cabo USB-Serial, excedeu o limite suportável de memória para o tamanho do arquivo implicando no estouro de memória e depois que houve este problema, o notebook e o software não reconhecia mais as portas USB.

Portanto a programação no microcontrolador foi realizada no software Terminal, linguagem em *Assembly*. Esta linguagem gerou um grau de dificuldade por ser uma linguagem pouco estudada. Após vários estudos desta linguagem o programa necessário para o funcionamento do protótipo foi realizado com sucesso.

A implementação do circuito para o acionamento da lâmpada foi outra dificuldade encontrada, pois o acionamento era para ter sido realizado com o foto acoplador MOC 3040 e um TRIAC 226, mas este dois circuitos não conseguiram fazer o acionamento da lâmpada devido a sua alta complexidade de implementação. Portanto para que o circuito funcionasse foi necessária a utilização de um foto acoplador PC 817 e um relé devido sua facilidade.

4.3. Sugestões para trabalho futuros

Neste projeto várias linhas de pesquisas que podem ser criadas, com fundamento no mesmo. Seguem abaixo sugestões de trabalho futuros para este projeto.

- Acionamento de alarmes.
- Controle de todos equipamentos elétricos encontrados em um ambiente residencial.
- Controle do acionamento dos equipamentos e circuitos residenciais.
- Monitorar o gasto de energia em ambiente comercial, como em uma academia.
- Utilizar um interruptor com o objetivo de parar o tempo estipulado pelo usuário.
- Acionamento dos circuitos elétricos em uma residência sem a presença do ser humano.

5. Referencias Bibliográficas

GIMENEZ, Salvador Pinillos. **Microcontroladores 8051: Teoria do hardware e do software; Aplicações em controle digital; Laboratório e simulação.** São Paulo: Pearson Education do Brasil Ltda, 2002.

MIZRAHI, Victorine Viviane. **Treinamento em Linguagem C.** Módulo 1. São Paulo: Pearson Makron Books, 1990.

NICOLOSI, Denys Emílio Campion. **Microcontrolador 8051 Detalhado.** 5ª Edição. São Paulo: Érica, 2004.

SOARES, Márcio José. **Microcontroladores PIC – 6ª parte, Varredura de teclados.** Revista Mecatrônica Fácil. São Paulo. Ano 2, nº 11, Julho-Agosto/2003.

_____. **Microcontroladores PIC – 7ª parte, Controle de displays LCD.** Revista Mecatrônica Fácil. São Paulo. Ano 2, nº 12, Setembro-Outubro /2003.

SILVA JÚNIOR, Vidal Pereira da. **Aplicações Prática do Microcontrolador 8051.** 11ª Edição. São Paulo: Érica, 2003.

ORDONEZ, Edward David Moreno. PENTEADO, Cesar Giacomini. SILVA, Alexandre César Rodrigues da. **Microcontroladores e FPGAs: Aplicações em Automação –** Novitec, 2005.

NICOLOSI, Denys Emilio Campion. BRONZERI, Rodrigo Barbosa. **Microcontrolador 8051 com linguagem C: prático e didático – família AT89S8252 Atmel – 1.ed.—** São Paulo: Érica, 2005.

Endereço de Internet.

http://www.mzeditora.com.br/artigos/mic_modernos.htm. Acessado em 22/01/2008

http://www.coinfo.cefetpb.edu.br/professor/leonidas/irc/apostilas/comun_serial.pdf,
Acessado em 02/02/2008

<http://www.guiadohardware.net/tutoriais/memoria-flash>, Acessado em 25/04/2008

<http://www.mobilidadetudo.com/2006/08/computao-invisvel-e-m-commerce.html>
Acessado em 28/04/2008.

<http://pt.wikipedia.org/wiki/Triac>, Acessado em 03/05/2008

http://www.coinfo.cefetpb.edu.br/professor/leonidas/irc/apostilas/comun_serial.pdf,
Acessado em 08/06/2008

<http://members.tripod.com/huilyrobot/compo/rele.htm>. Acessado em 08/06/2008

<http://www.burgoseletronica.net/fotoacopladores.htm>. Acessado em 05/06/2008

Apêndice.

```
.*****
;

.*****
;

.*****
;

.*****          *****
;

.*****  CENTRO UNIVERSITÁRIO DE BRASILIA          *****
;

.*****          UNICEUB          *****
;

.*****  ENGENHARIA DA COMPUTAÇÃO          *****
;

.*****  DISCIPLINA: PROJETO FINAL          *****
;

.*****
;

.*****  PROJETO: Circuito Residencial com Temporizador          *****
;

.*****          Individual por Usuário          *****
;

.*****
;

.*****  AUTORES: RICARDO REBELO SILVA MELO          *****
;

.*****  DATA INÍCIO: 21/04/2008          *****
;

.*****  DATA FINAL : DD/MM/AAAA          *****
;

.*****
;

.*****
;

.
;

.
;

$MOD51

$TITLE(TECLADO)

$PAGEWIDTH(132)

$DEBUG

$OBJECT

$NOPAGING

.
;

.
;
```



```

INICIO: CLR    P3.0          ;DESLIGA A LAMPADA

        LCALL  INILCD        ;CHAMA A ROTINA DO DISPLAY. INICIALIZA O DISPLAY
                                LCD

        LCALL  INIREL        ;CHAMA A ROTINA DE RELOGIO. INICIALIZA O RELOGIO

        LCALL  TEXTO1        ;CHAMA A ROTINA DE ESCRITA NO DISPLAY "PROJETO
                                FINAL"

        LCALL  SECLIN        ;CHAMA A ROTINA PARA ESCREVER NA SEGUNDA LINHA

        LCALL  TEXTO2        ;CHAMA A ROTINA DE ESCRITA NO DISPLAY
                                "TEMPORIZADOR"

        LCALL  RDTECL        ;CHAMA A ROTINA DE LEITURA DO TECLADO

        MOV    A,R3          ;ACUMULADOR RECEBE O VALOR DO REGISTRADOR.
                                RECUPERA O VALOR DA TECLA

        CJNE   A,#02AH,PULA1 ;ACUMULADOR RECEBE O VALOR DE '*', VAI PARA A
                                FUNÇÃO PULA1. SE TECLOU * = PEDIDO DE SENHA.

        LCALL  TEXTO3        ;CHAMA A ROTINA DE ESCRITA NO DISPLAY "DIGITE SUA
                                SENHA" NA LINHA 1.

        LCALL  SECLIN        ;CHAMA A FUNÇÃO DE ESCREVER NA SEGUNDA LINHA.]

        LCALL  SDEF          ;CHAMA A ROTINA DA SENHA DEFAULT.

        LCALL  SENHA         ;CHAMA A FUNÇÃO DE SENHA DIGITADA NO TECLADO.

```

LCALL COMPAR ;CHAMA A FUNÇÃO DE COMPARAR SENHA. VERIFICA SE
A SENHA DIGITADA É IGUAL A SENHA DEFAULT.

PULA1: JB F0,INICIO ;SE A SENHA ESTIVER ERRADA VOLTA PARA O INICIO

LCALL TEXTO4 ;CHAMA A ROTINA DE ESCRITA NO DISPLAY. "DIGITE O TEMPO."

LCALL RDTECL ;CHAMA A ROTINA DE LEITURA DO TECLADO.

MOV A,R3 ;ACUMULADOR RECEBE O VALOR DO REGISTRADOR.
RECUPERA O VALOR DA TECLA.

CJNE A,#023H,PULA2 ;ACUMULADOR RECEBE O VALOR DE '#', VAI PARA A
FUNÇÃO PULA2. SE TECLOU '#' = PEDIDO DE TEMPO

LCALL SECLIN ;CHAMA A ROTINA PARA ESCREVER NA SEGUNDA LINHA.

LCALL TEMPO ;CHAMA A FUNÇÃO TEMPO. LE A TEMPORIZAÇÃO
ATRAVES DO TECLADO

ESPERA: LCALL MOSTRA ;CHAMA A FUNÇÃO MOSTRA. VAI MOSTRAR O RELOGIO
NO DISPLAY

LCALL UMSEG ;CHAMA A FUNÇÃO UMSEG. DECREMENTA UM SEGUNDO.

LCALL TSTCLK ;CHAMA A FUNÇÃO DE TESTE O RELOGIO. TESTA SE
PASSOU A TEMPORIZAÇÃO.

PULA2: LJMP ESPERA ;CHAMA A FUNÇÃO ESPERA. FICA ESPERANDO A
TEMPORIZAÇÃO ACABAR.

.....
; ;

2

;

2

```
.....
```

```

;PREPARA LCD PARA RECEBER INSTRUÇÃO

```

```
;PREPARA PULSO DA ESCRITA PARA O DISPLAY
```

```
;PREPARA PARA LIMPAR O DISPLAY
```

```
;VÁ LIMPAR O DISPLAY LCD;
```

```
;PREPARA A PRIMEIRA POSIÇÃO DO DISPLAY
```

;VÁ LIMPAR O DISPLAY LCD;

```
;PREPARA PARA DESLOCAR PARA A DIREITA DO DISPLAY
```

```
;VÁ LIMPARA O DISPLAY LCD;
```

```
;PREPARA O CURSO APARENTE NO DISPLAY;
```

```
;VÁ LIMPAR O DISPLAY LCD;
```

```
;PREPARA DESLOCAMENTO PARA 8 BITS NO DISPLAY
```

```
;VÁ LIMPAR O DISPLAY LCD;
```

```
;ENVIA DADOS PARA O DISPLAY
```

```

RET                                ;RETORNA A ROTINA

.....
;
;
;   SUBROTINA DO LCD
;
;
;
.....

WRLCD: SETB  P1.0                ;PREPARA PARA RECEBER DADOS

LCALL  DELAY                    ;ESPEARA PASSAR O TEMPO DE XXX.

CLR    P1.0                    ;DESABILITA A ESCRITA

RET                                ;RETORNA A ROTINA

.....
;
;
;   ROTINA DE DELAY
;
;
;
.....

DELAY: MOV  R5,#0FFH            ;CARREGA REGISTRADOR DE 256 Useg

MOV  R6,#02H                  ;CARREGA REGISTRADOR MULTIPLEXADOR DE 2

DELAY1: DJNZ R5,$              ;ESPERA PASSAR 256 uSEG

```

DJNZ R6,DELAY1 ;ESPERA NOVAMENTE PASSAR 256 uSEG

RET ;RETORNA A ROTINA

.....
; ;

; INICIALIZAÇÃO DO RELOGIO ; ;
; ; ;

.....
; ;

INIREL: MOV R0,#020H ;APONTA PARA A POSIÇÃO 20H DA MEMORIA RAM.

INICILIZAÇÃO DO RELOGIO

MOV A,#030H ;ACUMULADOR INICIALIZADO COM 00 DO SEGUNDOS

MOV @R0,A ;ARMAZENA ACUMULADOR NA POSIÇÃO 20H

INC R0 ;INCREMENTA A POSIÇÃO DA MEMORIA DE 20H PARA 21H

MOV @R0,A ;ARMAZENA O ACUMULADOR NA POSIÇÃO 21H

INC R0 ;INCREMENTA A POSIÇÃO DA MEMORIA DE 21H PARA
22H

MOV A,#03AH ;NA POSIÇÃO 22H DA MEMORIA O ACUMULADOR VAI
RECEBER ':'

MOV @R0,A ;ARMAZENA O ACUMULADOR NA POSIÇÃO 22H

INC R0 ;INCREMENTA A POSIÇÃO DA MEMORIA DE 22H PARA 23H

```

MOV    A,#030H          ;ACUMULADOR INICIALIZA COM 00 DOS  MINUTOS

MOV    @R0,A             ;ARMAZENA ACUMULADOR NA POSIÇÃO 24H
INC    R0                 ;INCREMENTA DA POSIÇÃO DA MEMORIA 24H PARA 25H
MOV    @R0,A             ;ARMAZENA ACUMULADOR NA POSIÇÃO 25H
INC    R0                 ;INCREMENTA A POSIÇÃO DA MEMORIA DE 25H PARA 26H
MOV    A,#03AH          ;NA POSIÇÃO 25H DA MEMORIA O ACUMULADOR VAI RECEBER
!:'
MOV    @R0,A             ;ARMAZENA O ACUMULADOR NA POSIÇÃO 25H
INC    R0                 ;INCREMENTA A POSIÇÃO DA MEMORIA DE 25H PARA 26H
MOV    A,#030H          ;ACUMULADOR INICIALIZA COM 00 DAS HORAS
MOV    @R0,A             ;ARMAZENA ACUMULADOR NA POSIÇÃO 26H
INC    R0                 ;INCREMENTA DA POSIÇÃO DA MEMORIA 26H PARA 27H
MOV    @R0,A             ;ARMAZENA ACUMULADOR NA POSIÇÃO 27H
RET                                ;RETORNA A INIREL

.....
;                                     ;
;   ROTINA PARA ECREVER NO LCD       ;
;                                     ;
.....

TEXT01:    MOV    DPTR,#0600H      ;CHAMA POSIÇÃO DA TABELA TAB1
          TEXT01: "PROJETO FINAL"

LCALL  FRASE                      ;CHAMA A FUNÇÃO FRASE. ESCRIVE A FRASE
          NO DISPLAY

RET                                ;RETORNA

```

TEXT02: MOV DPTR,#0700H ;CHAMA A POSIÇÃO A TABELA TAB2 TEXT02:
"TEMPORIZAÇÃO".

LCALL FRASE ;CHAMA A FUNÇÃO FRASE. ESCRIVE A FRASE
NO DISPLAY

RET ;RETORNA

TEXT03: LCALL INILCD ;CHAMA A FUNÇÃO INICIALIZA LCD.

MOV DPTR,#0800H ;CHAMA A POSIÇÃO DA TABELA TAB3 TEXT03: "DIGITE SUA
SENHA".

LCALL FRASE ;CHAMA A FUNÇÃO FRASE. ESCRIVE A FRASE NO
DISPLAY

RET ;RETORNA

TEXT04: LCALL INILCD ;CHAMA A FUNÇÃO INICIALIZA LCD.

MOV DPTR,#0900H ;CHAMA A POSIÇÃO DA TABELA TAB4 TEXT04: "DIGITE O
TEMPO".

LCALL FRASE ;CHAMA A FUNÇÃO FRASE. ESCRIVE A FRASE NO
DISPLAY

RET ;RETORNA

TEXT05: LCALL INILCD ;CHAMA A FUNÇÃO INICIALIZA LCD.

MOV DPTR,#0A00H ;CHAMA A POSIÇÃO DA TABELA TAB5 TEXT05: "LIGA LED".

LCALL FRASE ;CHAMA A FUNÇÃO FRASE. ESCRIVE A FRASE NO
DISPLAY

RET ;RETORNA

```

TEXTO6: LCALL  INILCD          ;CHAMA A FUNÇÃO INICIALIZA LCD.

        MOV    DPTR,#0B00H      ;CHAMA A POSIÇÃO DA TABELA TAB6 TEXTO6: "DELIGA
LED".

        LCALL  FRASE            ;CHAMA A FUNÇÃO FRASE. ESCRIVE A FRASE NO
DISPLAY

        RET                      ;RETORNA

```

```

TEXTO7: LCALL  INILCD          ;CHAMA A FUNÇÃO INICIALIZA LCD.

        MOV    DPTR,#0C00H      ;CHAMA A POSIÇÃO DA TABELA TAB7 TEXTO7: "SENHA
INCORRETA".

        LCALL  FRASE            ;CHAMA A FUNÇÃO FRASE. ESCRIVE A FRASE NO
DISPLAY

        RET                      ;RETORNA

```

```

FRASE: MOV    A,#000H          ;ACUMULADOR RECEBE O VALOR 0.

        MOVC   A,@A+DPTR        ;ACUMULADOR RECEBE O ACUMULADOR COM DPTR

        MOV    P0,A             ;A PORTA P0 RECEBE O ACUMULADOR. NA PORTA P0 ESTÁ O
DISPLAY LCD

        LCALL  WRLCD            ;CHAMA A FUNÇÃO PARA ESCRIVER NO DISPLAY LCD.

        INC    DPL              ;VAI LER O PROXIMO CARACTER

        CJNE   A,#02EH,FRASE    ;LE ATÉ ACABAR A FRASE

VOLTA: RET                      ;RETORNA AO PROGAMA PRINCIPAL.

```

```

.....

```

```

;

```

```

;

```

```

;   ROTINA PARA ESCRIVER NA SEGUNDA LINHA DO DISPLAY   ;

```

```

;

```

```

;

```

```

.....

```

```

SECLIN: CLR   P1.1           ;PREPARA LCD PARA RECEBER INSTRUÇÃO
      CLR   P1.0           ;PREPARA PULSO DA ESCRITA PARA O DISPLAY
      MOV   P0,#0C0H       ;ESCREVE NA PROXIMA LINHA
      LCALL WRLCD          ;VÁ LIMPAR O DISPLAY LCD;
      MOV   P0,#006H       ;PREPARA PARA DESLOCAR PARA A DIREITA DO DISPLAY
      LCALL WRLCD          ;VÁ LIMPARG O DISPLAY LCD;
      MOV   P0,#00EH       ;PREPARA O CURSO APARENTE NO DISPLAY;
      LCALL WRLCD          ;VÁ LIMPARG O DISPLAY LCD;
      MOV   P0,#038H       ;PREPARA DESLOCAMENTO PARA 8 BITS NO DISPLAY
      LCALL WRLCD          ;VÁ LIMPARG O DISPLAY LCD;
      SETB  P1.1           ;ENVIA DADOS PARA O DISPLAY
      RET                  ;RETORNA

```

```

.....

```

```

;
;
;   ROTINA DE LEITURA DO TECLADO
;

```

```

.....

```

```

RDTECL: CLR   F0           ;INICIALIZA A FLAG      RDTECL: LEITURA DE TECLADO
      LCALL TECL           ;CHAMA A FUNÇÃO DO TECLADO
      JNB   F0,RDTECL       ;SE NÃO TECLOU CONTINUA LENDO O TECLADO
      RET                  ;RETORNA.

```

```

.....

```

```

;
;
;   ROTINA DO TECLADO MATRICIAL 4X3
;
;

```

```

.....

```

```

TECL: CLR    F0                ;INICIALIZA A FLAG

      MOV    R3,#0FFH          ;DEFINE O NÚMERO DE COLUNAS DO TECLADO

      MOV    R4,#04H           ;DEFINE O NÚMERO DE LINHAS DO TECLADO

      MOV    A,#0FEH           ;PREPARA O ACUMULADOR PARA INICIALIZAR O P2


PRXLIN: MOV   P2,A             ;INICIALIZA O P2

      INC    R3                ;INCREMENTA NO REGISTRADOR

      JNB    P2.4,TECLOU       ;COLUNA 1. TESTA O PRIMEIRO NÚMERO DA LINHA 1, LINHA 2,
LINHA 3 E LINHA 4 DO TECLADO.

      INC    R3                ;INCREMENTA O R3. VERIFICA SE A TECLA PRECIONADA NA
COLUNA 1

      JNB    P2.5,TECLOU       ;COLUNA 2. TESTA O SEGUNDO NÚMERO DA LINHA 1, LINHA2,
LINHA3 E LINHA 4 DO TECLADO.

      INC    R3                ;INCREMENTA O R3. VERIFICA SE A TECLA PRECIONADA NA
COLUNA 2

      JNB    P2.6,TECLOU       ;COLUNA 3. TESTA O TERCEIRO NÚMERO DA LINHA 1, LINHA 2,
LINHA 3 E LINHA 4 DO TECLADO.

      RL     A                 ;ROTEA O ACUMULADOR PARA PRÓXIMA CASA

      DJNZ   R4,PRXLIN         ;DECREMENTA E PULA SE OS VALORES NA LINHAS 1, LINHA 2,
LINHA 3 E LINHA 4 NÃO FOR ZERO. SE O VALOR NA LINHA FOR ZERO PULA A PROXIMA
LINHA

      RET                     ;RETORNA A ROTINA


TECLOU: SETB   F0              ;ATIVA O FLAG

      MOV    A,R3              ;O REGISTRADOR RECEBE O ACUMULADOR

      LCALL  TECLA             ;CHAMA A ROTINA TECLA

      MOV    R3,A              ;O ACUMULADOR RECEBE O REGISTRADOR

```



```

.....
;
;
;   ROTINA DE DEBOUNCE. ROTINA RESPONSÁVEL PARA QUE QUANDO UMA TECLA ;
;   DO TECLADO FOR SETADA ELA LER SOMENTE DEPOIS QUE A TECLA FOR SOLTA. ;
;   PARA QUE NÃO OCORRA A LEITURA INCORRETA. ;
;
;
.....

```

```

PARADA: MOV    A,P2          ;LER P2 PARA EXECUTAR A PARADA
          CJNE   A,#0FEH,TESTE1      ;SE NÃO SOLTOU A TECLA ESPERA SOLTAR
          LCALL  DELAY              ;CHAMA A ROTINA DE DELAY. EXECUTA A PARADA
          RET                               ;RETORNA A PARADA

```

```

TESTE1: CJNE   A,#0FDH,TESTE2      ;SE NÃO SOLTOU A TECLA ESPERA SOLTAR
          LCALL  DELAY              ;CHAMA A ROTINA DE DELAY. EXECUTA A PARADA
          RET                               ;RETORNA A PARADA

```

```

TESTE2: CJNE   A,#0FBH,TESTE3      ;SE NÃO SOLTOU A TECLA ESPERA SOLTAR
          LCALL  DELAY              ;CHAMA A ROTINA DE DELAY. EXECUTA A PARADA
          RET                               ;RETORNA A PARADA

```

```

TESTE3: CJNE   A,#0F7H,PARADA      ;SE NÃO SOLTOU A TECLA ESPERA SOLTAR
          LCALL  DELAY              ;CHAMA A ROTINA DE DELAY. EXECUTA A PARADA
          RET                               ;RETORNA A PARADA

```

```

TECLA: MOV    DPTR,#0500H ;CHAMA A FUNÇÃO DPRT PARA A POSIÇÃO 05
          MOVC  A,@A+DPTR          ;ACUMULADOR RECEBE ACUMULADOR COM DPTR

```

```

RET                                ;RETORNA

.....

;                                ;

;   ROTINA DA SENHA DEFAULT      ;

;                                ;

.....

SDEF: MOV   R7,#004H              ;REGISTRADOR RECEBE 4 CARACTERS

      MOV   R0,#028H              ;REGISTRADOR VAI PARA A POSIÇÃO 28H DA MEMORIA RAM

      MOV   A,#031H              ;ACUMULADOR RECEBE O VALOR 1

OUTRO: MOV   @R0,A                ;REGISTRADOR RECEBE O ACUMULADOR

      INC   R0                    ;INCREMENTA O REGISTRADOR INDO PARA A POSIÇÃO 29H DA
MEMORIA RAM

      INC   A                    ;INCREMENTA O ACUMULADOR

      DJNZ  R7,OUTRO              ;REGISTRADOR PULA E CHAMA A OUTRA FUNÇÃO OUTRO

      RET                        ;RETORNA

.....

;                                ;

;   ROTINA DA SENHA              ;

;                                ;

.....

SENHA: MOV   R7,#004H              ;O REGISTRADOR RECEBE 4 POSIÇÕES NA MEMORIA
SENHA: DEFINE A SENHA

      MOV   R0,#02CH              ;INICIO DA SENHA DIGITADA NA POSIÇÃO 2C DA RAM
INTERNA

```

.....
;;

```

;
;
;   ROTINA DE LEITURA DA SENHA
;
;
```

.....
;;

```
RDKEY: CLR   F0           ;INICIALIZA A FLAG   RDKEY: ESCREVER A SENHA NO
DISPLAY

      LCALL  TECL          ;CHAMA A FUNÇÃO DO TECLADO

      JNB    F0,RDKEY      ;SE NÃO TECLOU CONTINUA LENDO O TECLADO

      MOV    A,R3          ;ACUMULADOR RECEBE O REGISTRADOR DO TECLADO

      MOV    @R0,A         ;ACUMULADOR VERIFICAR OS DADOS DO
REGISTRADOR

      INC    R0            ;INCREMENTA O REGISTRADOR

      DJNZ   R7,RDKEY      ;O REGISTRADOR DECREMENTA E PULA PARA A FUNÇÃO DE
ESCRITA DE SENHA

      RET                ;RETORNA
```

.....
;;

```

;
;
;   ROTINA DE COMPRA A SENHA DEFAULT COM A SENHA DIGITADA
;
;
```


; ROTINA DE LEITURA DA TEMPORIZAÇÃO DIGITADA ;

;

.....
.....

TEMPO: MOV R2,#003H ;REGISTRADOR RECEBE 3 CARACTERES.PREPARA
PARA ESCREVER NN:

 MOV R0,#020H ;REGISTRADOR RECEBE ESPAÇO EM BRNACO .INICIO
DA RAM NO RELOGIO

TEMPO1: MOV R7,#002H ;REGISTRADOR RECEBE 2 CARACTERES.QUANTIDADE DE
DITO DE CADA PASSO DO RELOGIO

TEMPO2: CLR F0 ;INICIALIZA A FLAG

 LCALL TECL ;CHAMA A FUNÇÃO DO TECLADO

 JNB F0,TEMPO2 ;FLAG PULA PARA A FUNÇÃO DO TEMPO2.SE NÃO TECLIU
CONTINUA TECLANDO.

 MOV A,R3 ;ACUMULADOR RECEBE O VALOR DO REGISTRADOR DO
TECLADO

 MOV @R0,A ;ACUMULADOR VERIFICA OS DADOS DO REGISTRADOR

 MOV P0,A ;PORTA P0 RECEBE OS DADOS DO ACUMULADOR. PREPARA
ESCREVE NÚMERO DIGITADO NO TECLADO

 LCALL WRLCD ;CHAMA A FUNÇÃO DE ESCREVER NO DISPLAY LCD

 INC R0 ;INCREMENTA O REGISTRADOR PARA A PROXIMA POSIÇÃO DA
MEMÓRIA

 DJNZ R7,TEMPO2 ;DECREMENTA O REGISTRADOR E PULA PARA A FUNÇÃO A
TEMPO2. SE NÃO ESCRVER NN VAI FINALIZAR

 MOV @R0,#03AH ;REGISTRADOR RECEBE O ':'.
 MOV P0,#03AH ;PREPARA PARA ESCREVER ':' NO DISPLAY LCD

 MOV P0,#03AH ;PREPARA PARA ESCREVER ':' NO DISPLAY LCD

 LCALL WRLCD ;CHAMA A FUNÇÃO DE ESCREVER NO DISPLAY.
ESCREVE ':' NO DISPLAY

 INC R0 ;INCREMENTA O REGISTRADOR. LE A PROXIMA POSIÇÃO DA
MEMORIA

DJNZ R2,TEMPO1 ;DECREMENTA O REGISTRADOR E PULA PARA A FUNÇÃO
TEMPO1.REINICIALIZA A ESCRITA DO PROXIMO NN:

TEMPO3: CLR F0 ;INICIALIZA A FLAG

LCALL TECL ;CHAMA A FUNÇÃO DO TECLADO

JNB F0,TEMPO3 ;FLAG PULA PARA A FUNÇÃO DO TEMPO2.SE NÃO TECLIU
CONTINUA TECLANDO.

MOV A,R3 ;ACUMULADOR RECEBE O VALOR DO REGISTRADOR DO
TECLADO

CJNE A,#023H,PULA3 ;ACUMLUDAOR TESTA SE TECLOU '#' = CONFIRMAÇÃO DE
TEMPORIZAÇÃO

PULA3: RET ;RETORNA

.....
))

;
;
; ROTINA QUE MOSTRA O RELOGIO NO DISPLAY ;
;
;

.....
))

MOSTRA: LCALL INILCD ;CHAMA A ROTINA DE INICIALIZAÇÃO DO DISPLAY LCD

MOV R4,#008H ;REGISTRADOR RECEBE 8 CARACTERES

MOV R1,#020H ;REGISTRADOR VAI PARA A POSIÇÃO 20H.

SHOW: MOV A,@R1 ;ACUMULADOR RECEBE DADOS DO REGISTRADOR.

MOV P0,A ;A PORTA P0 RECEBE OS DADOS DO ACUMULADOR.

LCALL WRLCD ;CHAMA A FUNÇÃO DE ESCRITA NO DISPLAY.

INC R1 ;INCREMENTA O REGISTRADOR. VAI PARA A PROXIMA
POSIÇÃO DA MEMORIA RAM


```

TSTCLK: MOV    R0,#027H          ;REGISTRADOR VAI PARA A POSIÇÃO 27H. APONTA A
UNIDADE DE SEGUNDO

        MOV    R1,#027H          ;REGISTRADOR VAI PARA A POSIÇÃO 27H. APONTA A
UNIDADE DE SEGUNDO

        CJNE   @R0,#30H,UNISEG    ;CONTINUA DECREMENTANDO SE NÃO CHEGOU AOS
60S. 3A É :

        MOV    @R1,#39H

        DEC    R0

        DEC    R1                ;DECREMENTA O REGISTRADOR. APONTA PARA A UNIDADE DE
MINUTOS

        CJNE   @R0,#30H,DEZSEG    ;CONTINUA DECREMENTANDO SE NÃO CHEGOU AOS
60S. 36H É 6

        MOV    @R1,#035H

        DEC    R0                ;PULA OS DOIS PONTOS

        DEC    R0                ;APONTA PARA A UNIDADE DE MINUTOS

        DEC    R1

        DEC    R1

        CJNE   @R0,#30H,UNIMIN    ;CONTINUA DECREMENTANDO SE NÃO CHEGAOU AOS 60M.
3A É :

        MOV    @R1,#39H

        DEC    R0                ;DECREMENTA O REGISTRADOR. APONTA PARA A UNIDADE
DE HORAS

        DEC    R1

        CJNE   @R0,#30H,DEZMIN    ;CONTINUA DECREMENTANDO SE NÃO CHEGOU AOS
60M. 36H É 6

        MOV    @R1,#035H

        DEC    R0                ;PULA OS DOIS PONTOS

        DEC    R0                ;APONTA A UNIDADE DE HORAS.

        DEC    R1

        DEC    R1

```


CJNE @R0,#30H,UNIHOR ;CONTINUA DECREMENTANDO SE NÃO CHEGOU AOS
60S. 36H É :

MOV @R1,#39H

DEC R0 ;DECREMENTA O REGISTRADOR. APONTA PARA A UNIDADE DE
HORAS

DEC R1

CJNE @R0,#30H,DEZHOR ;CONTINUA DECREMENTANDO SE NÃO CHEGOU AOS
60S. 36H É :

LJMP PISKA ;LOOP E PULA PARA A FUNÇÃO PISKA

UNISEG: DEC @R0 ;DECREMENTA UNIDADE DE SEGUNDO

RET

DEZSEG: DEC @R0 ;DECREMENTA DEZENA DE SEGUNDO

DEC R1

DEC R1

RET

UNIMIN: DEC @R0 ;DECREMENTA DEZENA DE HORA

DEC R1

RET

DEZMIN: DEC @R0 ;DECREMENTA DEZENA DE HORA

DEC R1

DEC R1

RET

UNIHOR: DEC @R0 ;DECREMENTA UNIDADE DE HORA

DEC R1

RET

DEZHOR: DEC @R0 ;DECREMENTA DEZENA DE HORA

DEC R1

DEC R1

RET

```
.....
;
;
;   ROTINA DE SINCRONISMO ENTRE O PISCAR DO LED E AS FRASES NO DISPLAY;
;
;
.....

PISKA: LCALL TEXTO5      ;ESCREVE A FRASE "LIGA O LED."

        SETB  P3.0      ;LIGA O LED

        LCALL UMSEG      ;ESPERA UM SEGUNDO COM O LED LIGADO

        LCALL UMSEG      ;ESPERA UM SEGUNDO COM O LED LIGADO

        LCALL TEXTO6     ;ESCREVE A FRASE "DESLIGA O LED."

        CLR   P3.0      ;DESLIGA O LED

        LCALL UMSEG      ;ESPERA UM SEGUNDO COM O LED DESLIGADO

        LCALL UMSEG      ;ESPERA UM SEGUNDO COM O LED DESLIGADO

        LCALL TECL      ;VÁ LER O TECLADO

        MOV   A,R3      ;RECUPERA VALOR DA TECLA

        CJNE  A,#02AH,PISKA ;TESTA SE TECLOU * = PEDIDO DE REINICIALIZACAO

        LJMP  INICIO     ;FICA AQUI PISCANDO O LED

.....

;
;
;   ROTINA DE TABELAS.
;
;
;
;
.....

ORG 0500H

TAB0: db '1' ;
      db '2' ;
```

```
db    '3'    ;  
db    '4'    ;  
db    '5'    ;  
db    '6'    ;  
db    '7'    ;  
db    '8'    ;  
db    '9'    ;  
db    '*'    ;  
db    '0'    ;  
db    '#'    ;
```

```
ORG    0600H
```

```
TAB1: db    ' PROJETO FINAL .';
```

```
ORG    0700H
```

```
TAB2: db    ' TEMPORIZADOR .';
```

```
ORG    0800H
```

```
TAB3: db    'DIGITE SUA SENHA.';
```

```
ORG    0900H
```

```
TAB4: DB    ' DIGITE O TEMPO .';
```

```
ORG    0A00H
```

```
TAB5: DB    ' LIGAR A LAMPADA.';
```

```
ORG    0B00H
```

```
TAB6: DB    ' DESLIGA LAMPADA.';
```

ORG 0C00H

TAB7: DB 'SENHA INCORRETA .';

END